Understanding the Structure of Large, Diverse Collections of Shapes

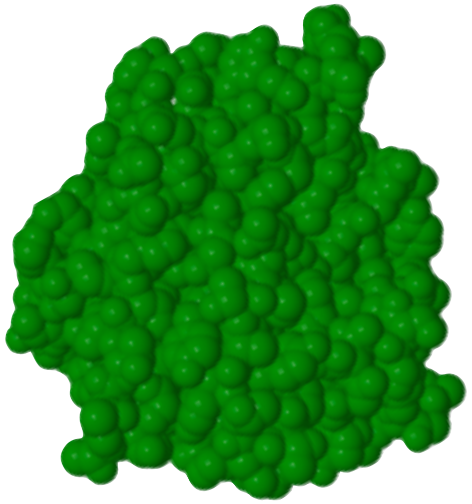
Vladimir G. Kim

Adviser: Thomas A. Funkhouser

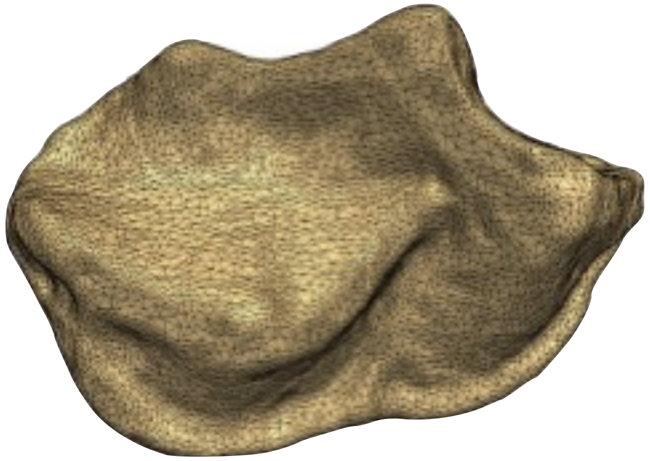
Princeton
University

Introduction

3D repositories



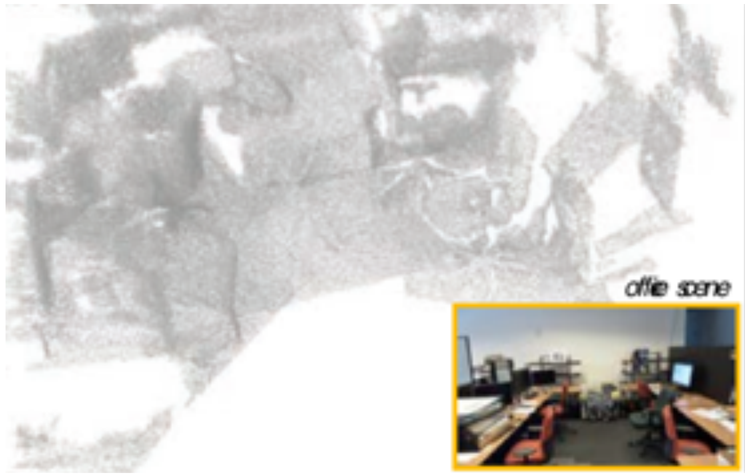
Molecular Biology



Paleonthology

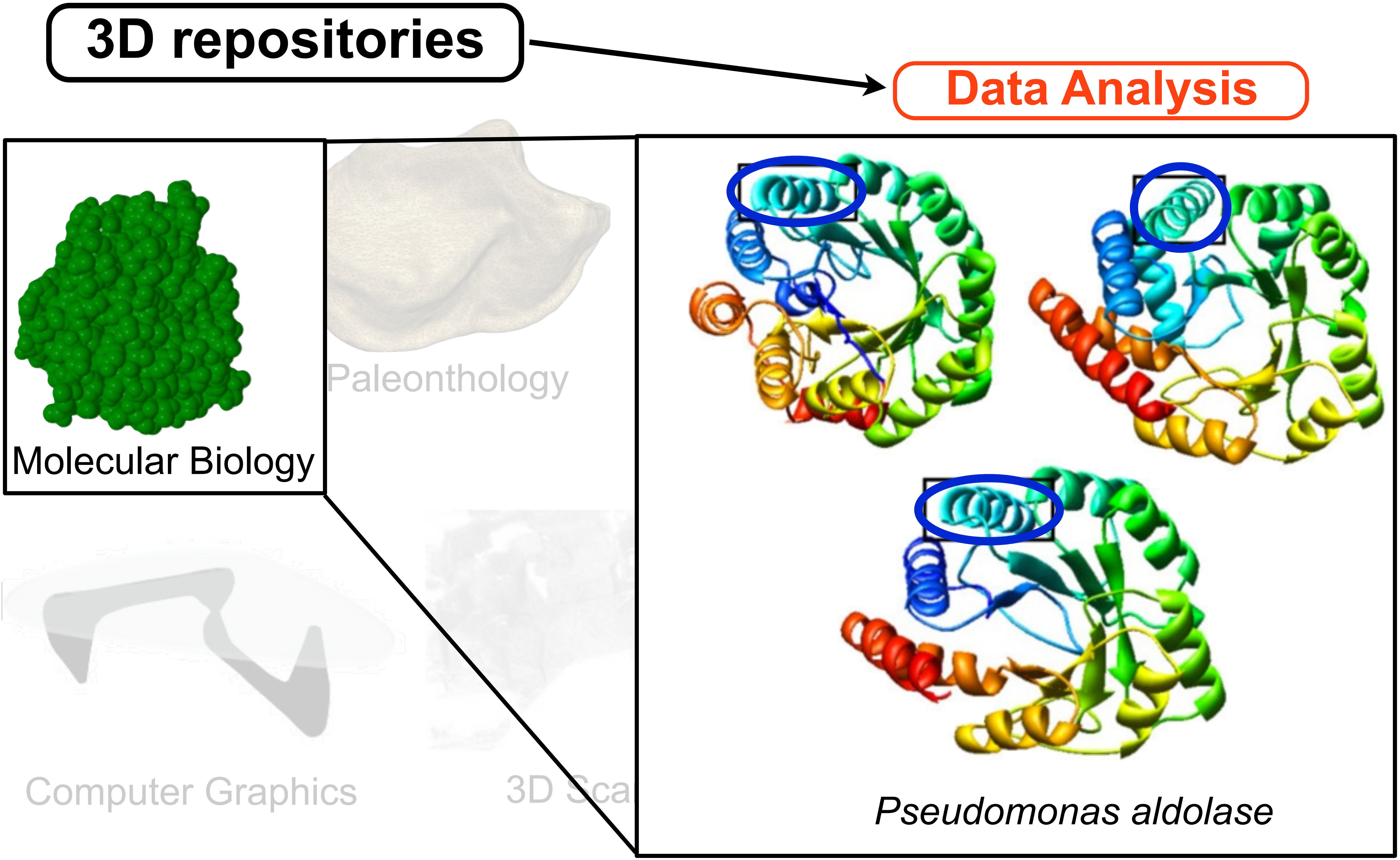
Medicine

Computer Graphics



3D Scanning

Introduction



3D repositories

Data Analysis

Molecular Biology

Paleonthology

Computer Graphics

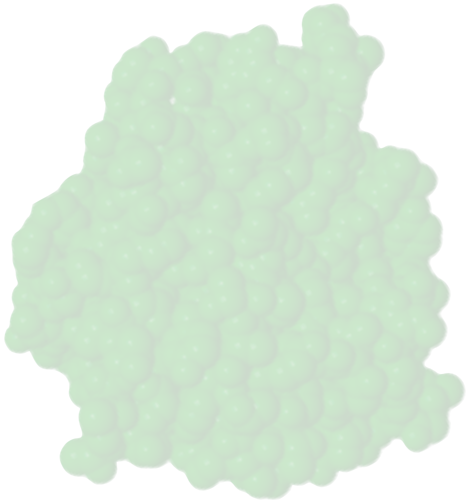
3D Sca

Pseudomonas aldolase

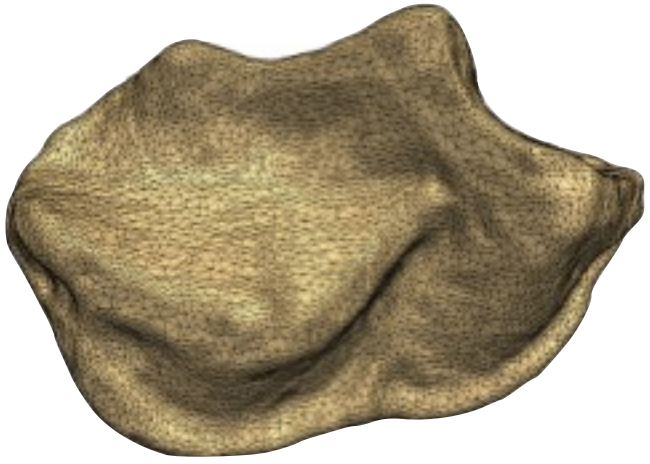
Introduction

3D repositories

Data Analysis



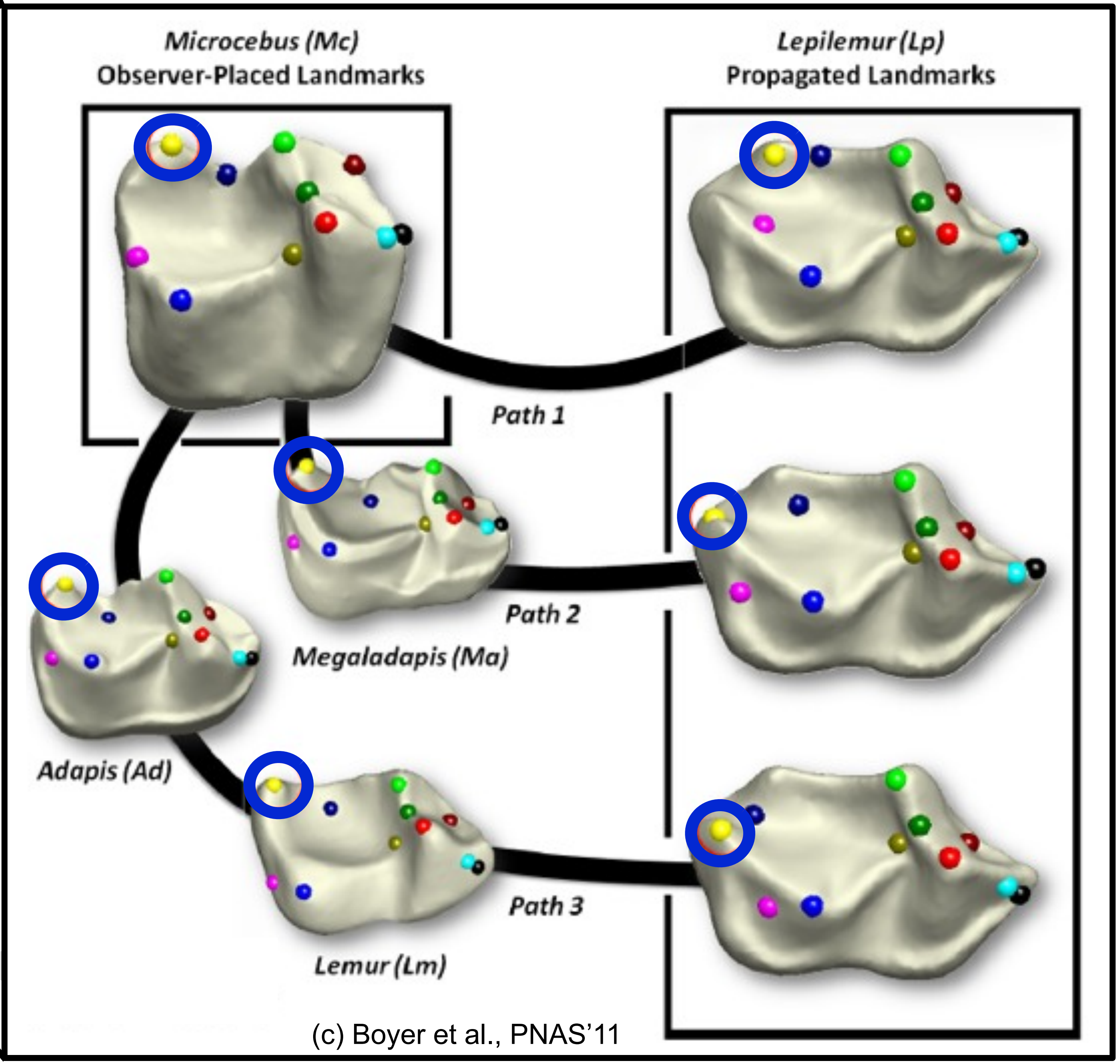
Molecular Biology



Paleonthology

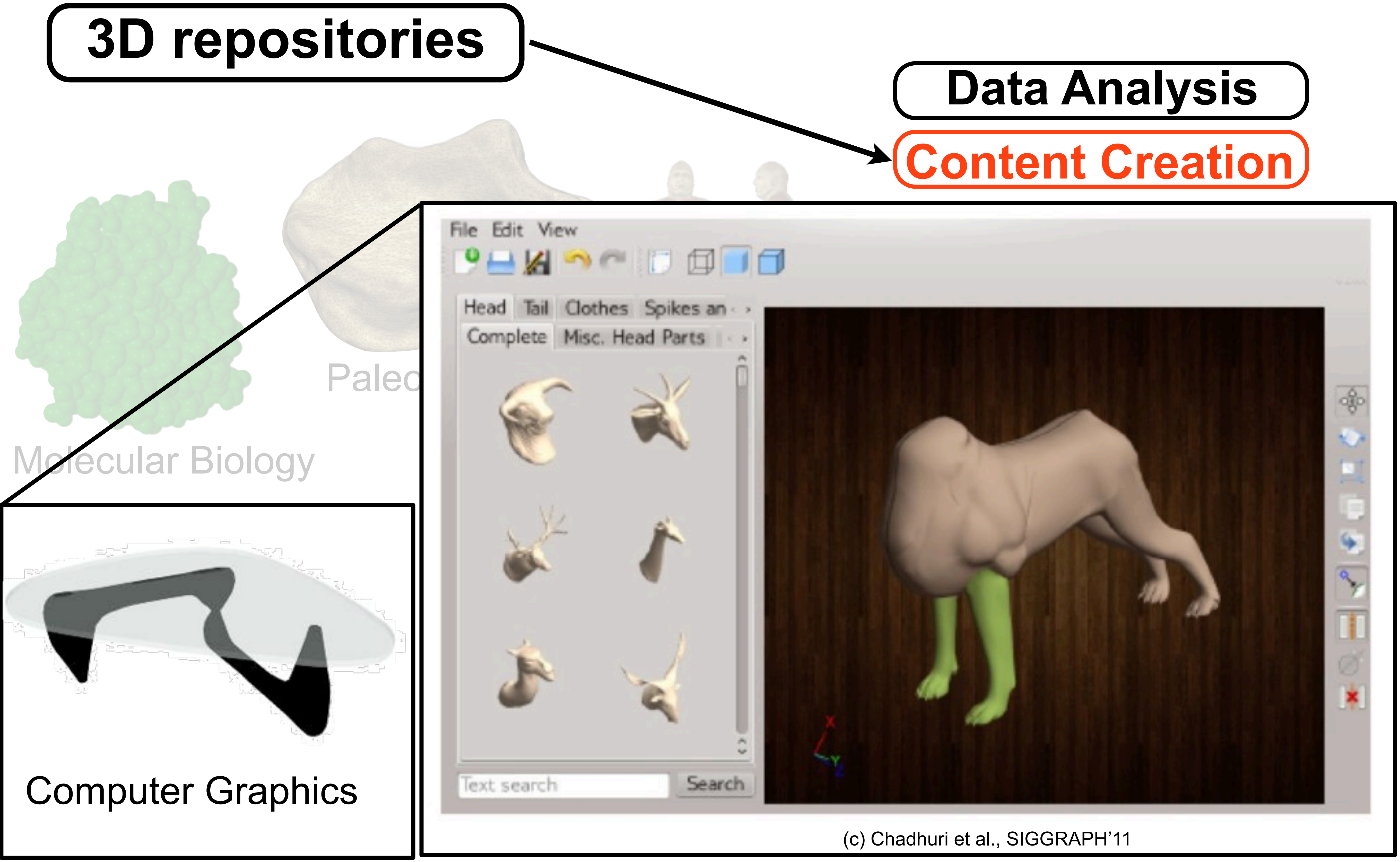
Computer Graphics

3D Sca



(c) Boyer et al., PNAS'11

Introduction



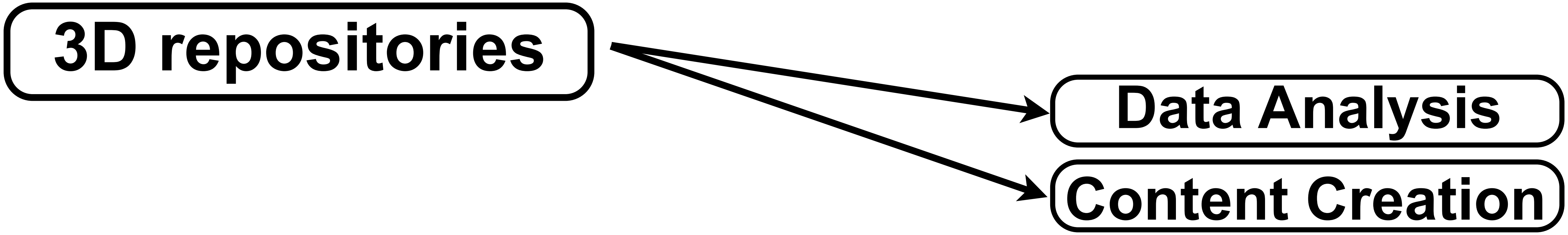
3D repositories

Data Analysis

Content Creation

(c) Chadhuri et al., SIGGRAPH'11

Introduction

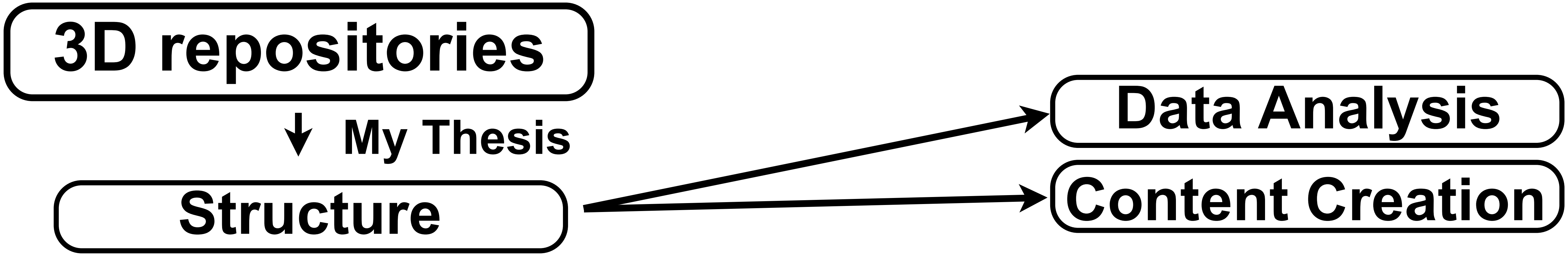


3D repositories

Data Analysis

Content Creation

Introduction



3D repositories

My Thesis

Structure

Data Analysis

Content Creation

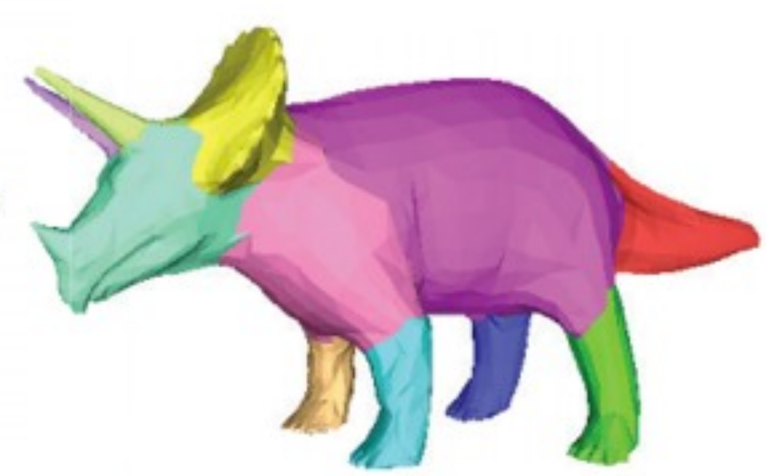
Previous Work

Understanding structure

- ➡ 3D shapes
- ○ Collections of 3D shapes

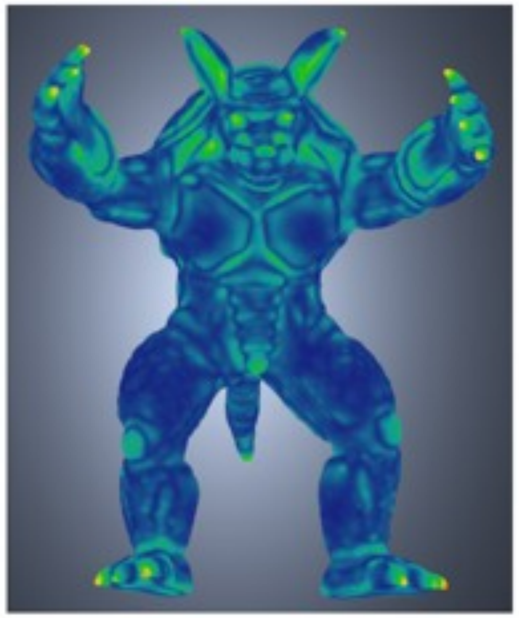
(c) Mitra et al., SIGGRAPH'06

Symmetry



(c) Golovinskiy et al. SIGGRAPH Asia'08

Segmentation



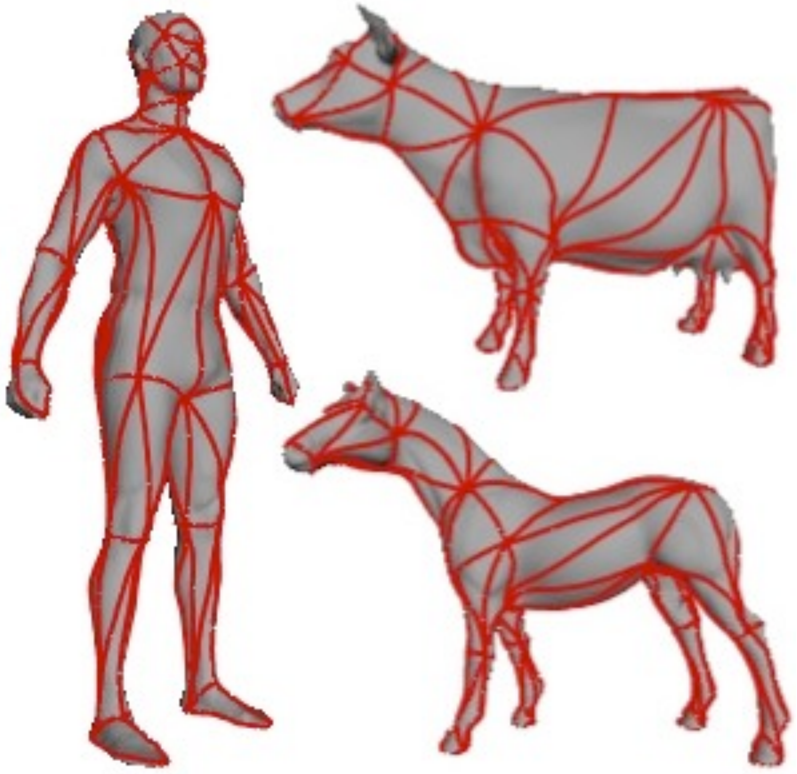
(c) Lee et al. SIGGRAPH'05

Saliency detection

Previous Work

Understanding structure

- o 3D shapes
- ➡ Collections of 3D shapes

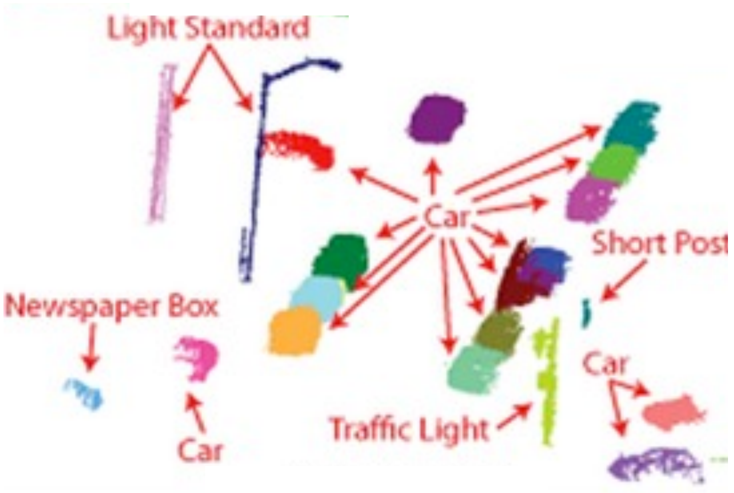


(c) Praun et al., SIGGRAPH'01

Correspondence

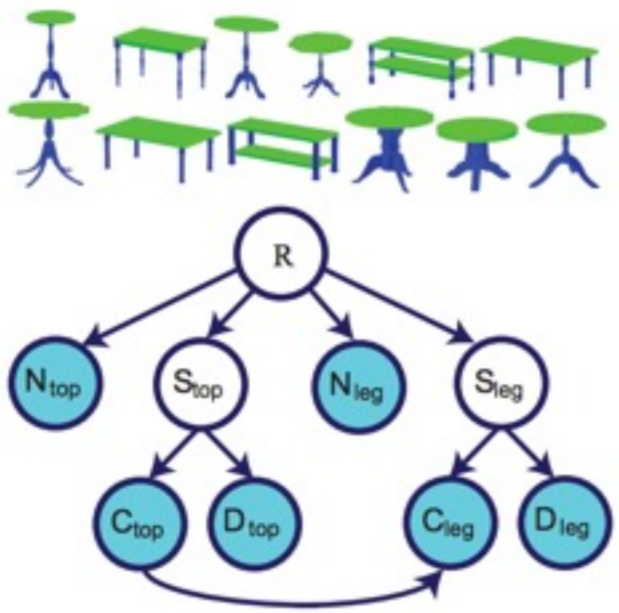
(c) Golovinskiy et al., SMI'09

Consistent Segmentation



(c) Golovinskiy et al., ICCV'09

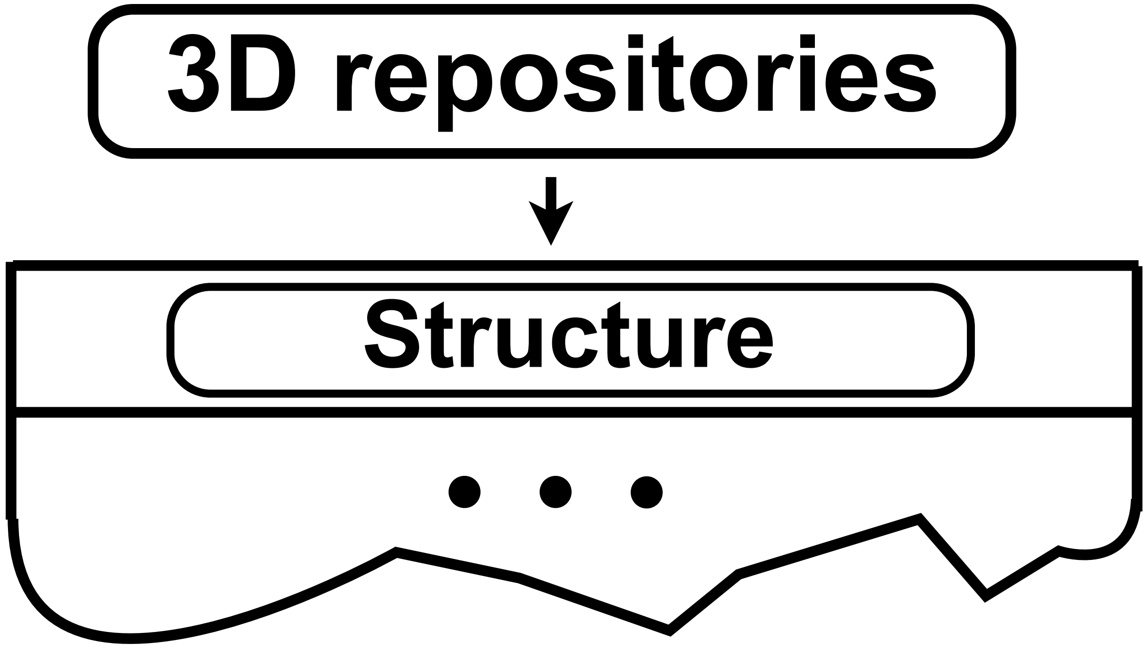
Grouping



(c) Kalogerakis et al. SIGGRAPH'12

Part-based models

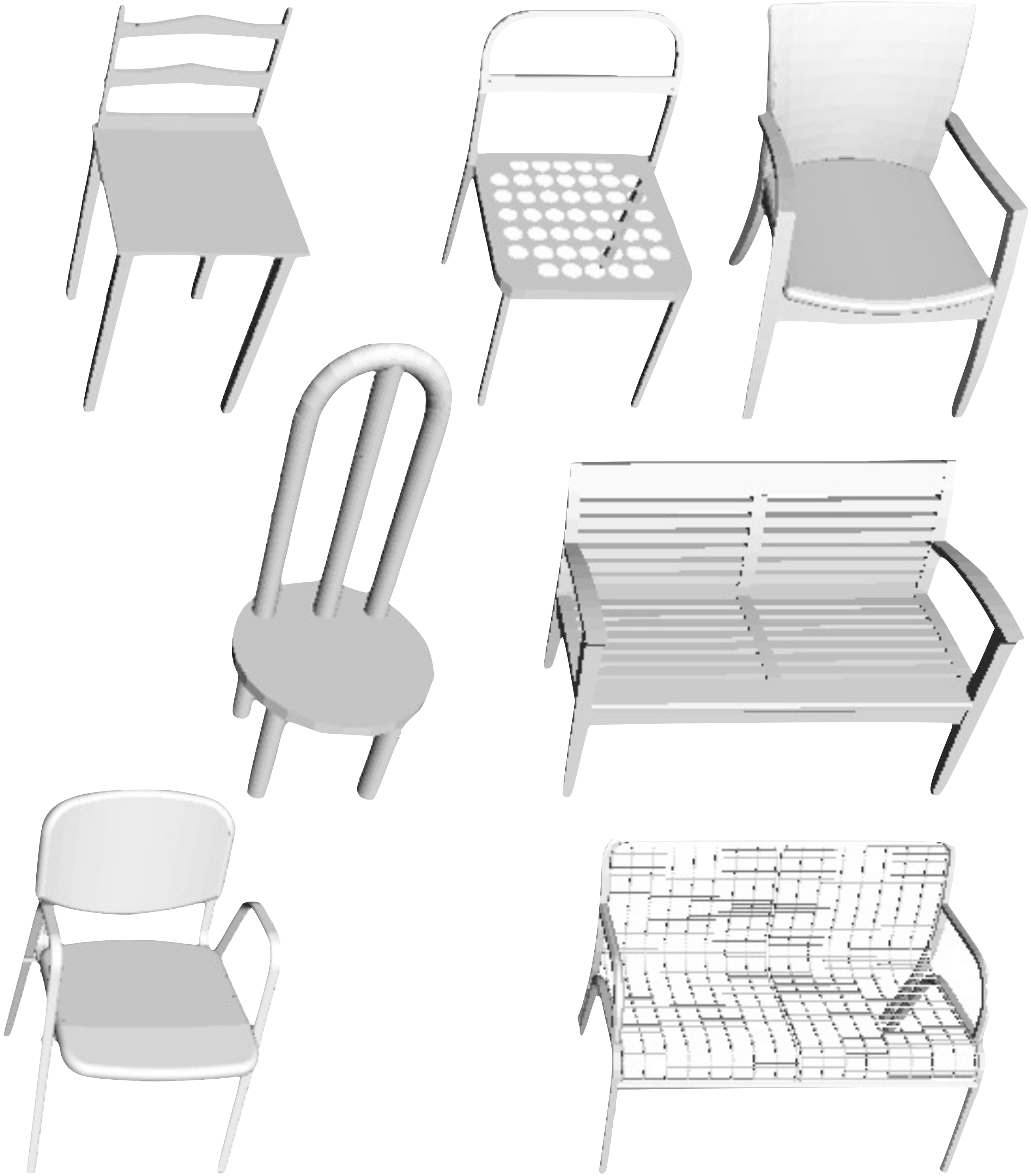
Goal



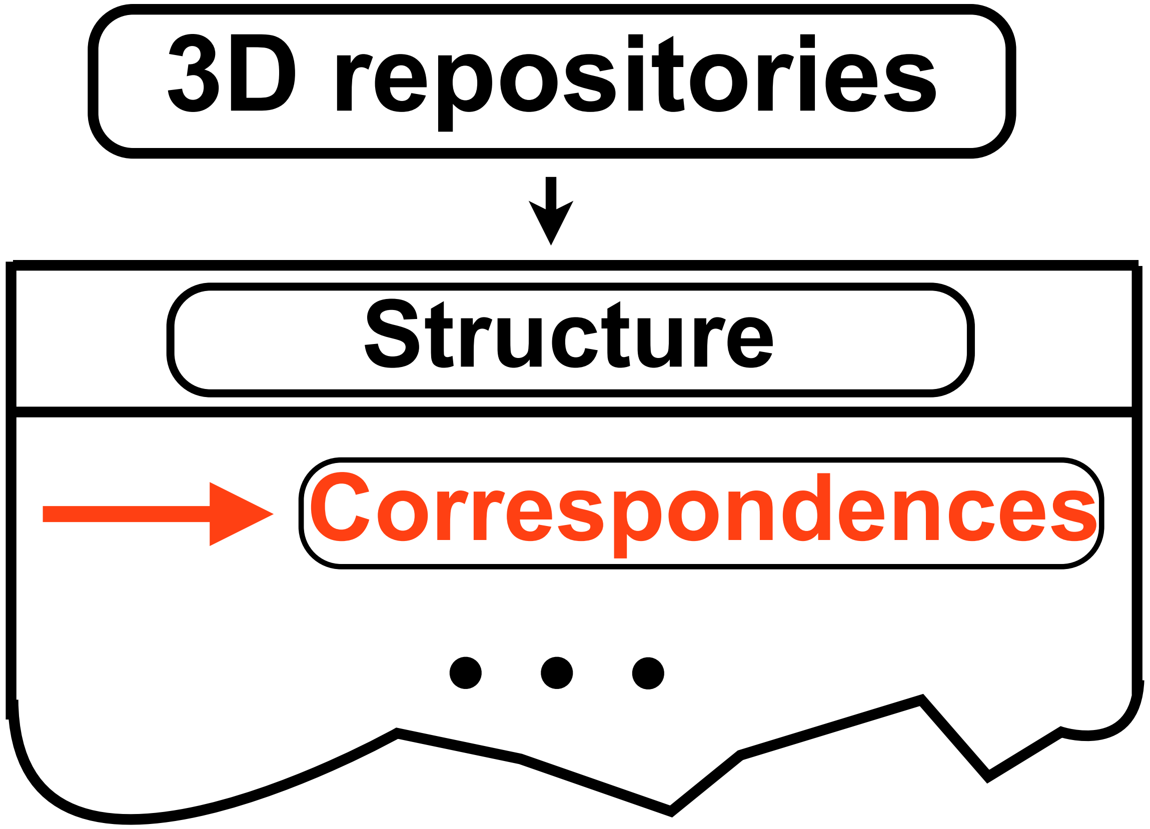
3D repositories

Structure

• • •



Goal

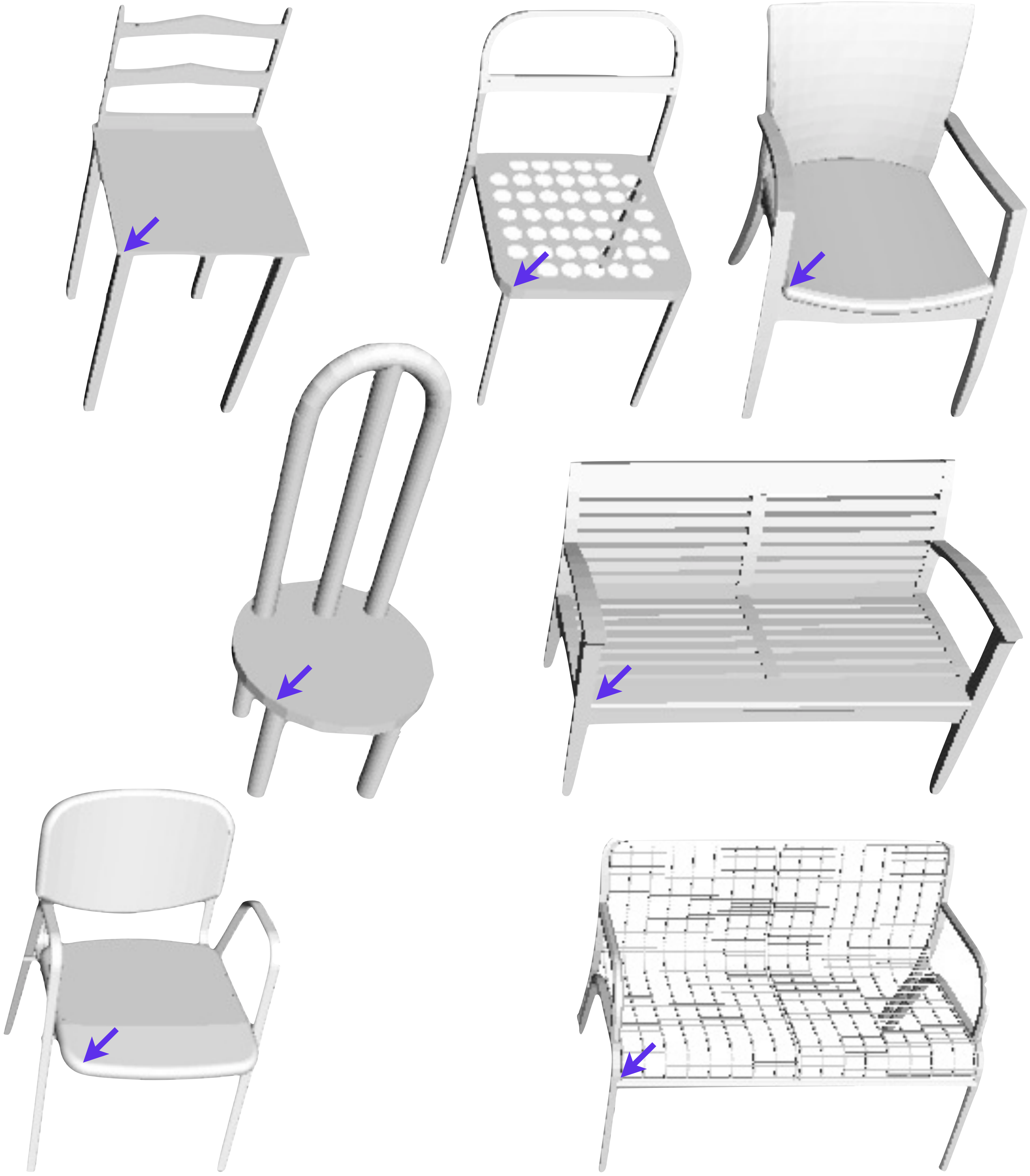


3D repositories

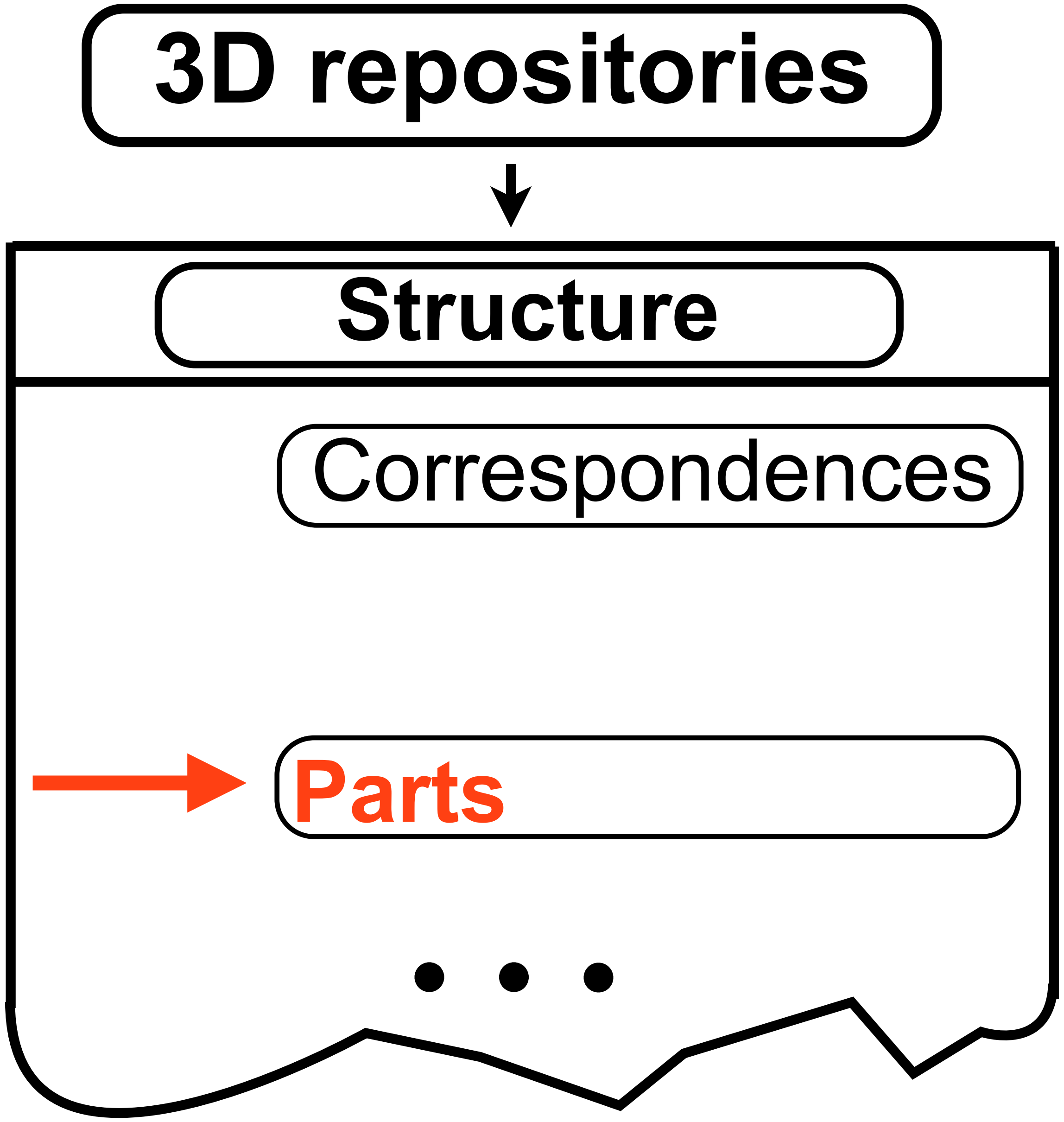
Structure

Correspondences

• • •



Goal



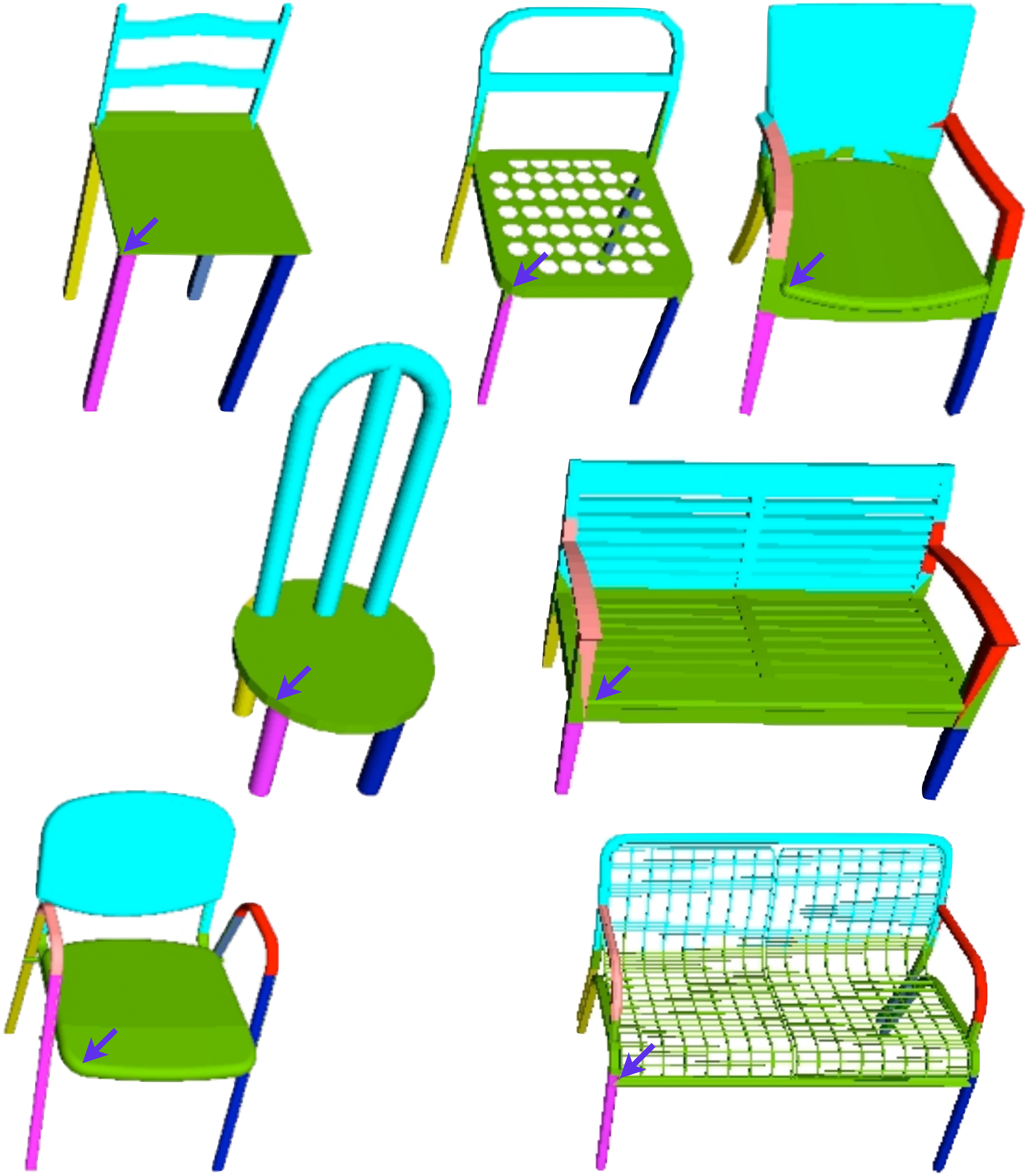
3D repositories

Structure

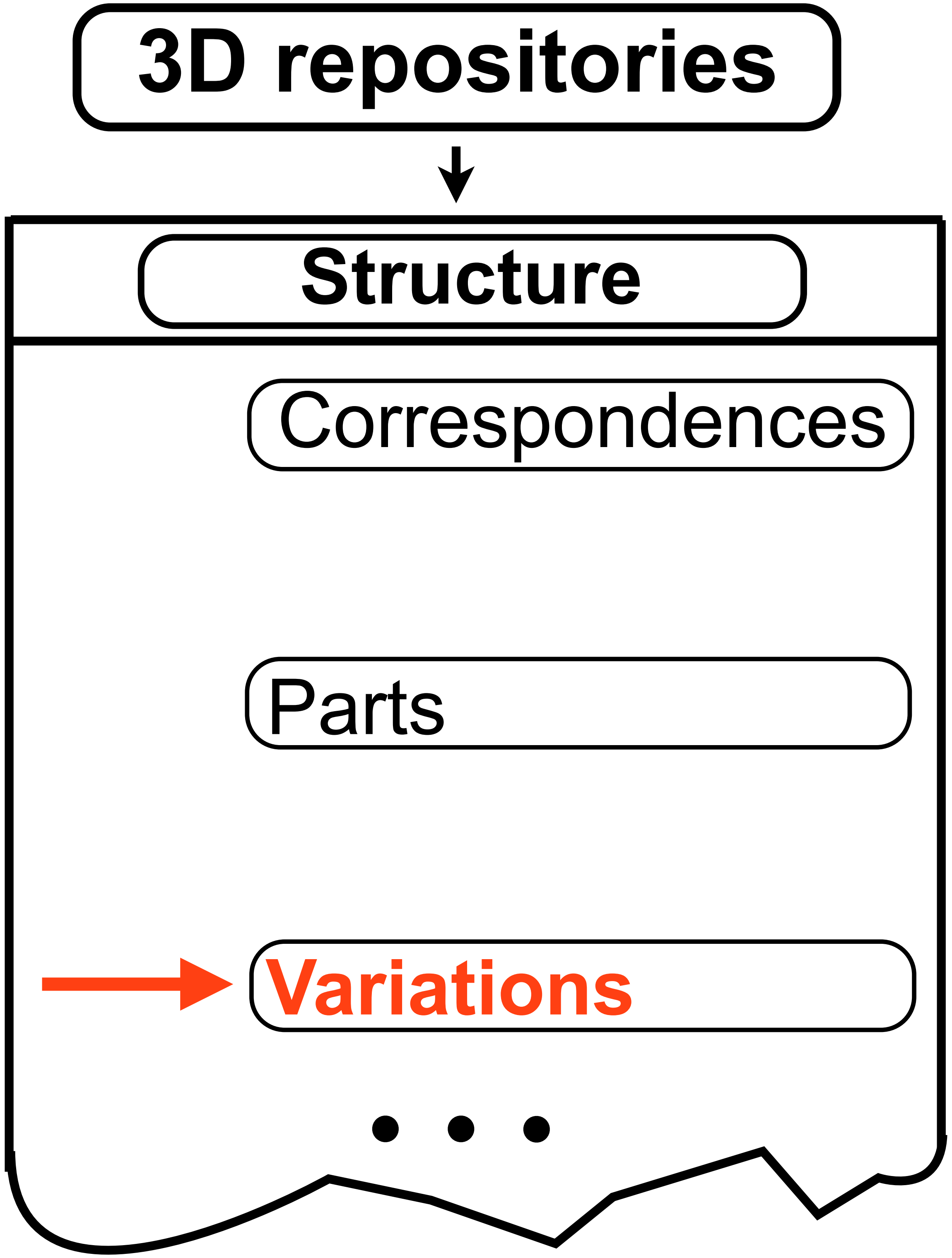
Correspondences

Parts

• • •



Goal



3D repositories

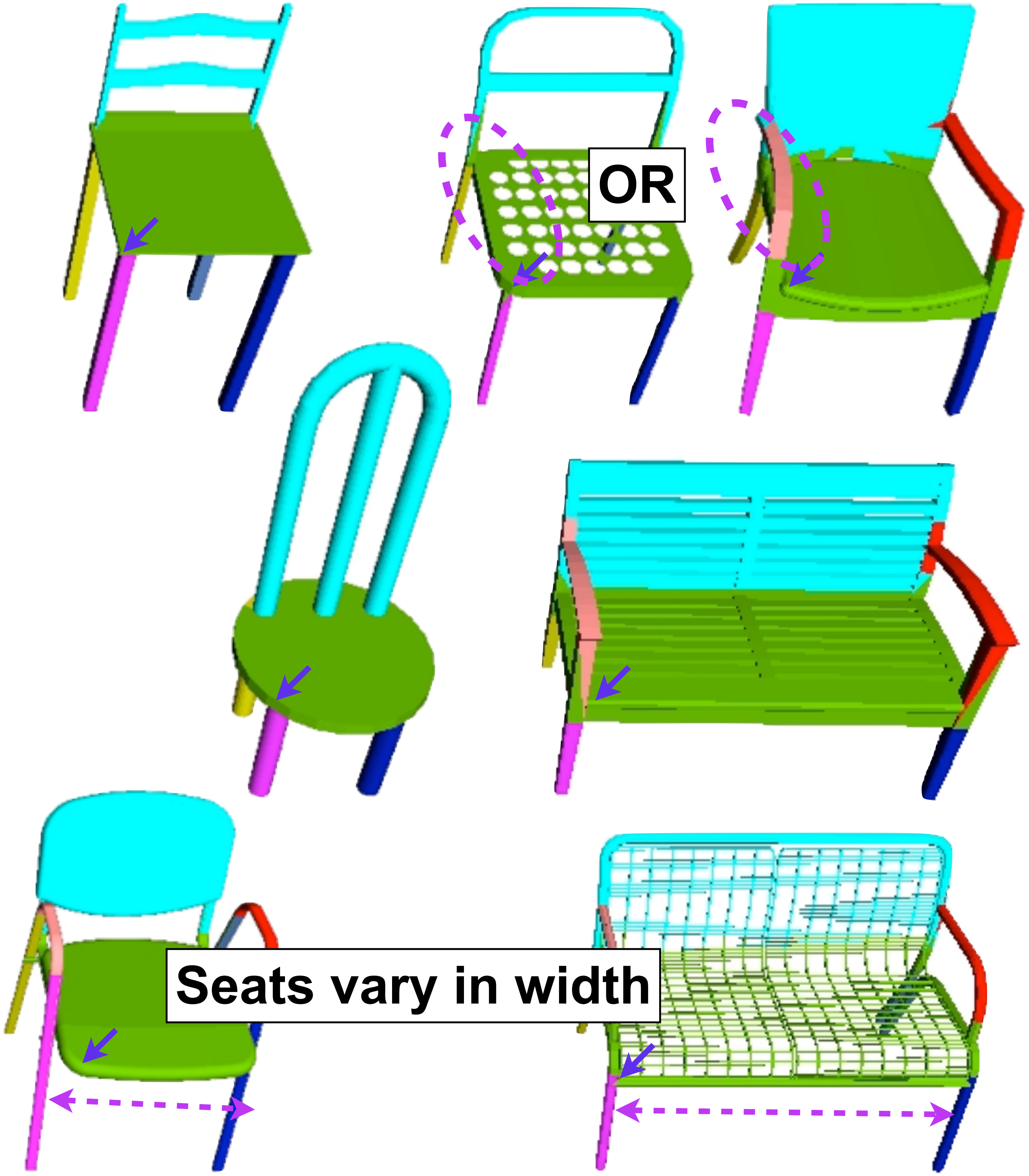
Structure

Correspondences

Parts

Variations

• • •



Goal

3D repositories

↓

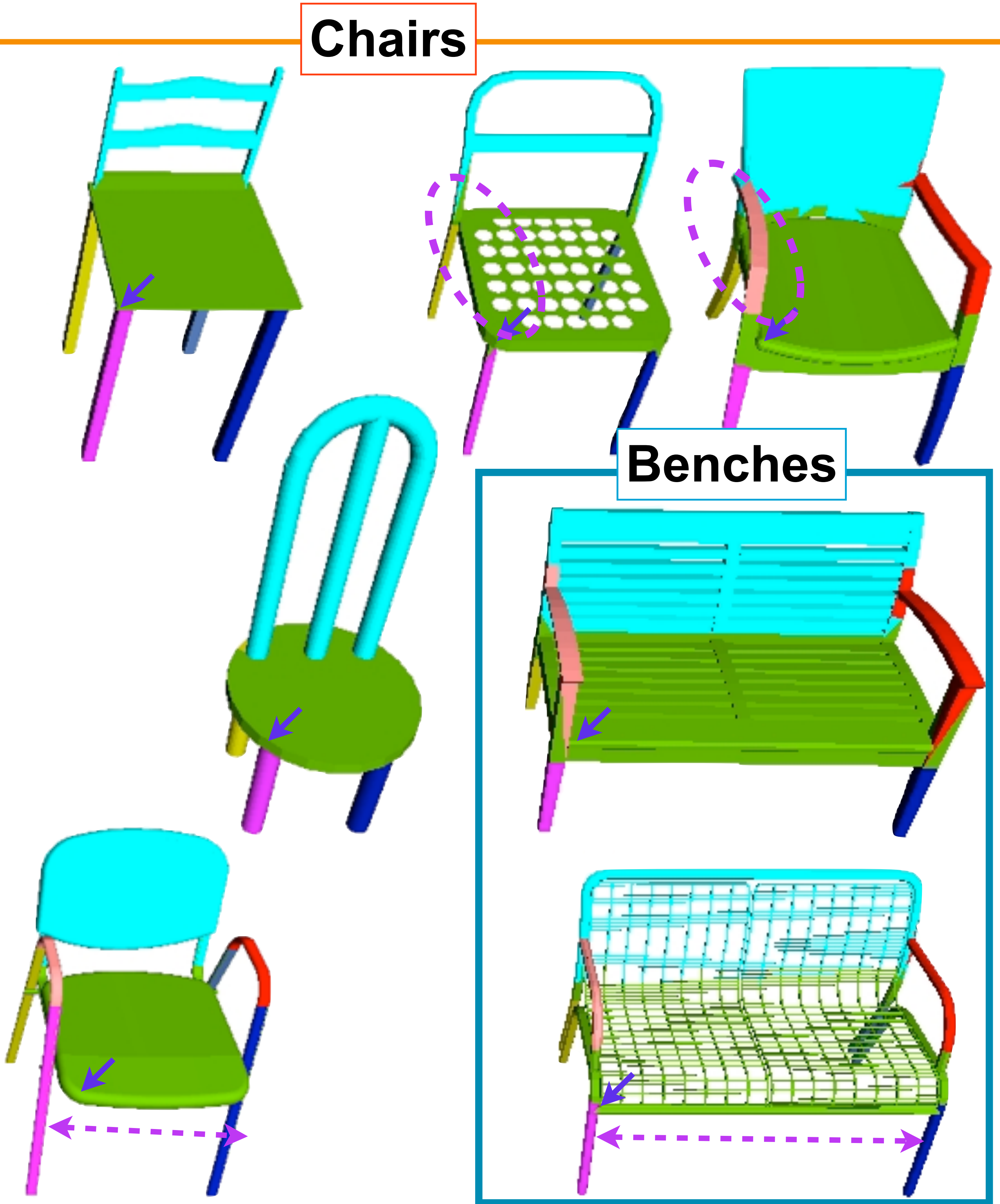
Structure

Correspondences

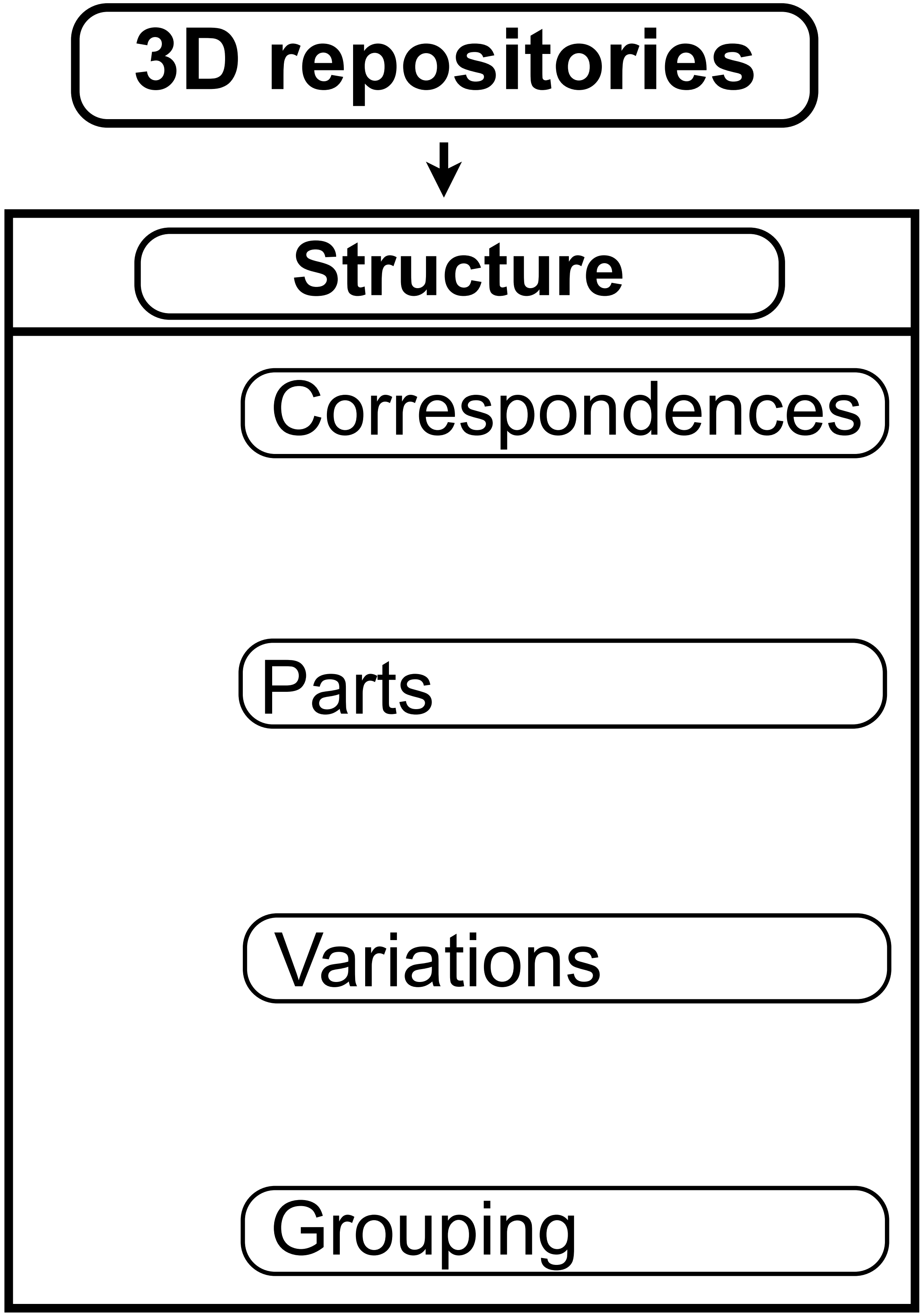
Parts

Variations

Grouping



Goal



3D repositories

Structure

Correspondences

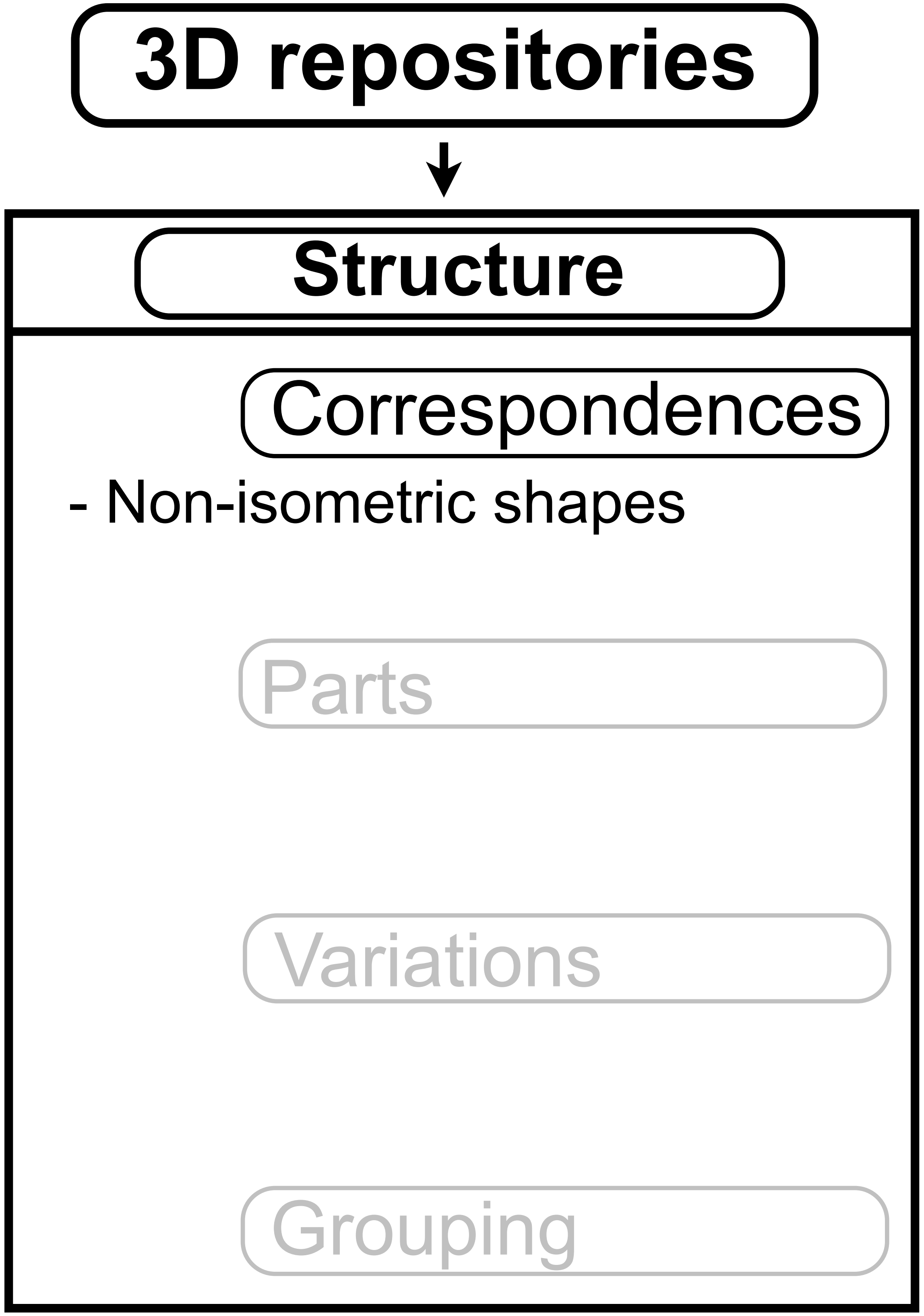
Parts

Variations

Grouping

1. Blended Intrinsic Maps
2. Fuzzy Correspondences
3. Deformable Template

Goal



3D repositories

Structure

Correspondences

- Non-isometric shapes

Parts

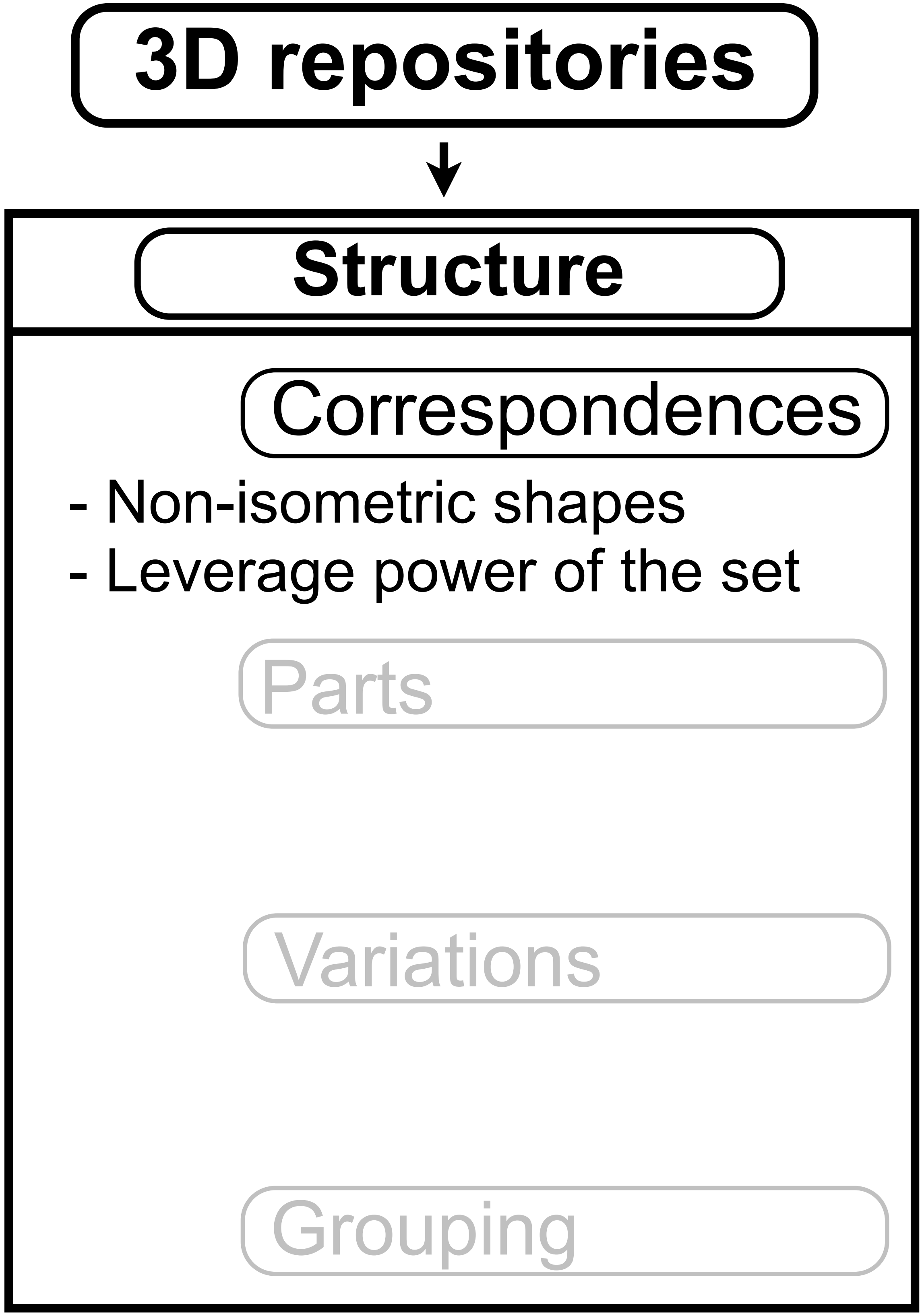
Variations

Grouping

1. **Blended Intrinsic Maps**
2. Fuzzy Correspondences
3. Deformable Template

Complexity: $O(N^2)$

Goal



3D repositories

Structure

Correspondences

- Non-isometric shapes
- Leverage power of the set

Parts

Variations

Grouping

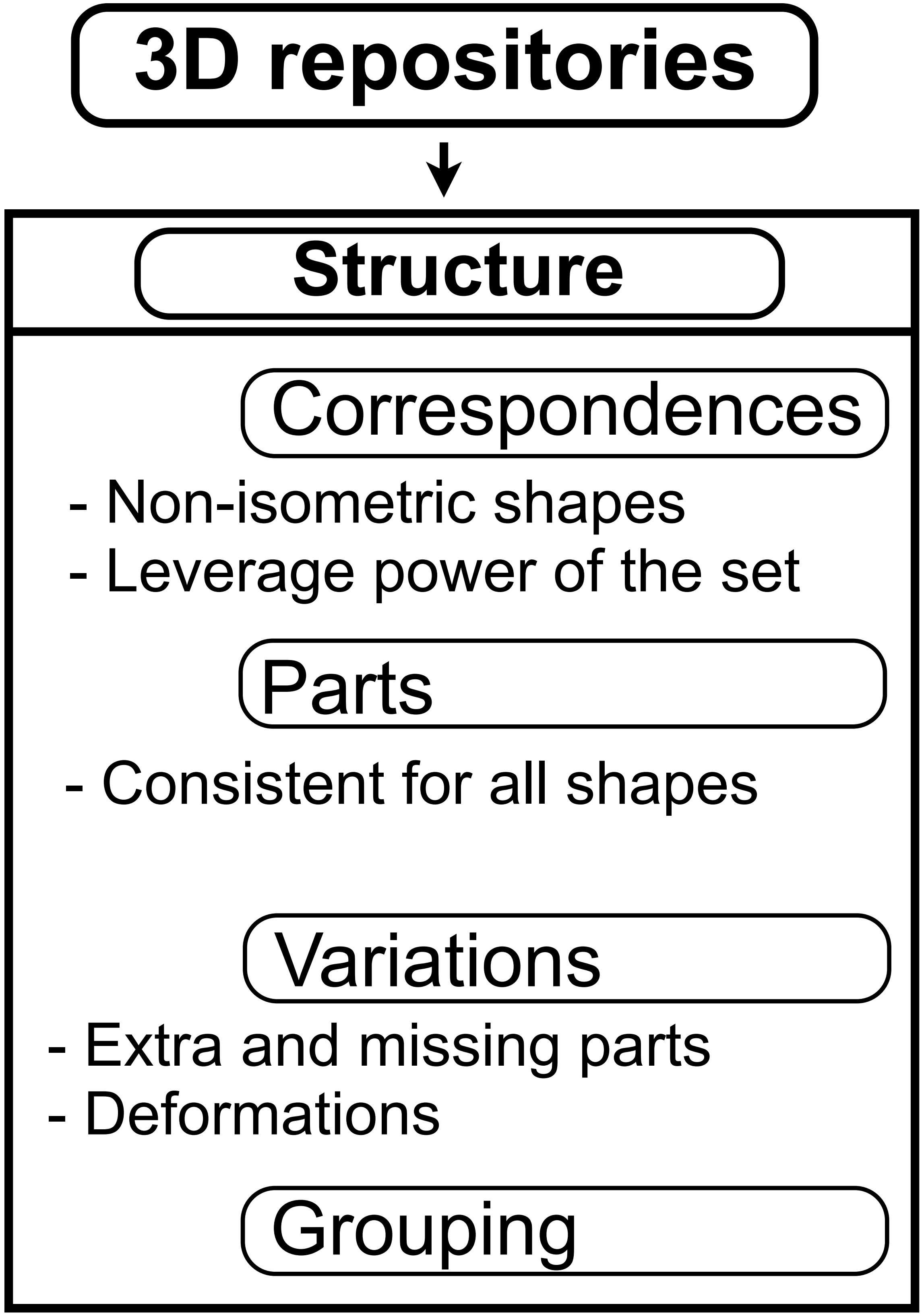
1. Blended Intrinsic Maps

2. Fuzzy Correspondences

3. Deformable Template

Complexity: $O(N^{1.5})$

Goal



3D repositories

Structure

Correspondences

- Non-isometric shapes
- Leverage power of the set

Parts

- Consistent for all shapes

Variations

- Extra and missing parts
- Deformations

Grouping

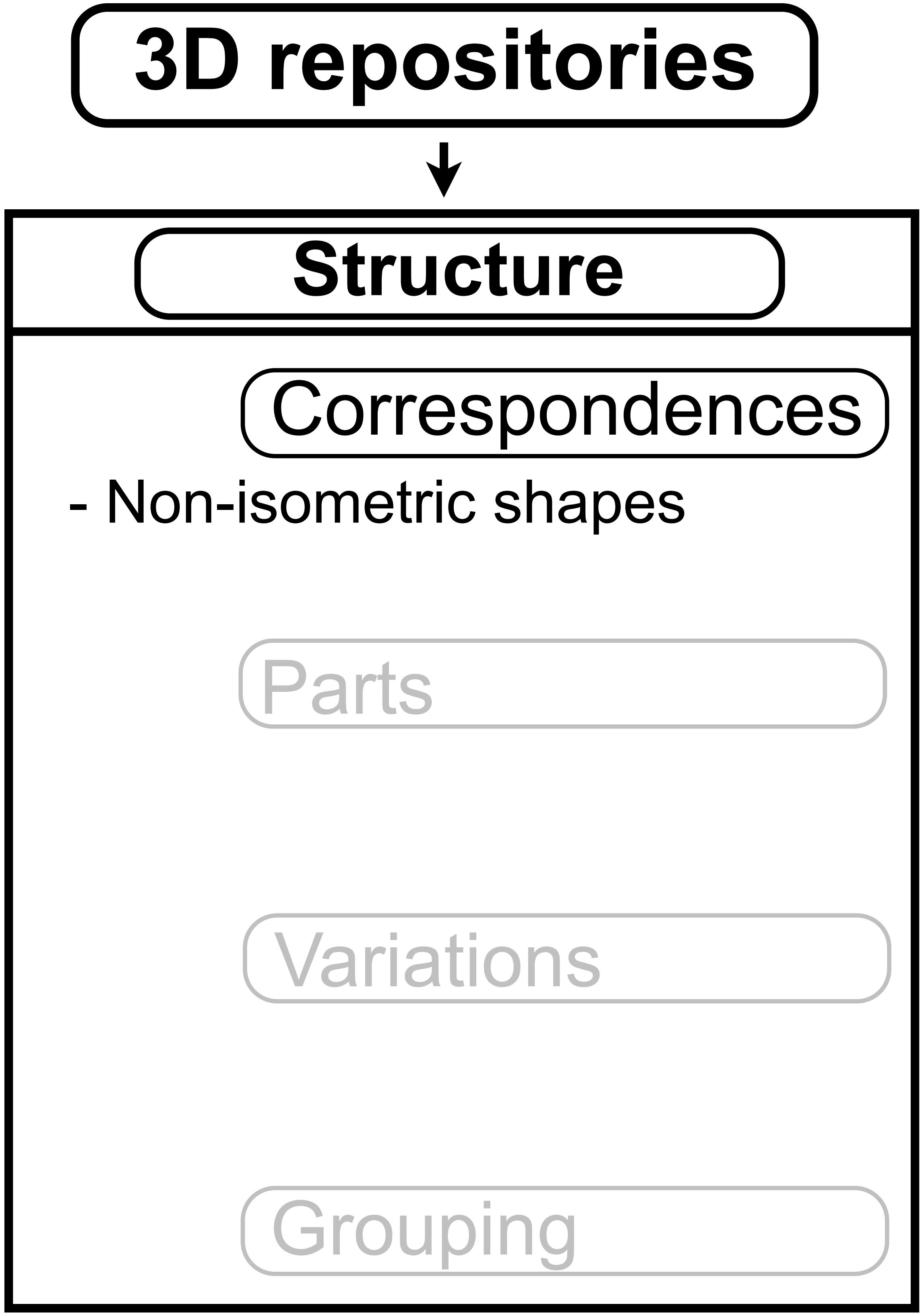
1. Blended Intrinsic Maps

2. Fuzzy Correspondences

3. Deformable Template

Complexity: O(N)

Goal



3D repositories

Structure

Correspondences

- Non-isometric shapes

Parts

Variations

Grouping

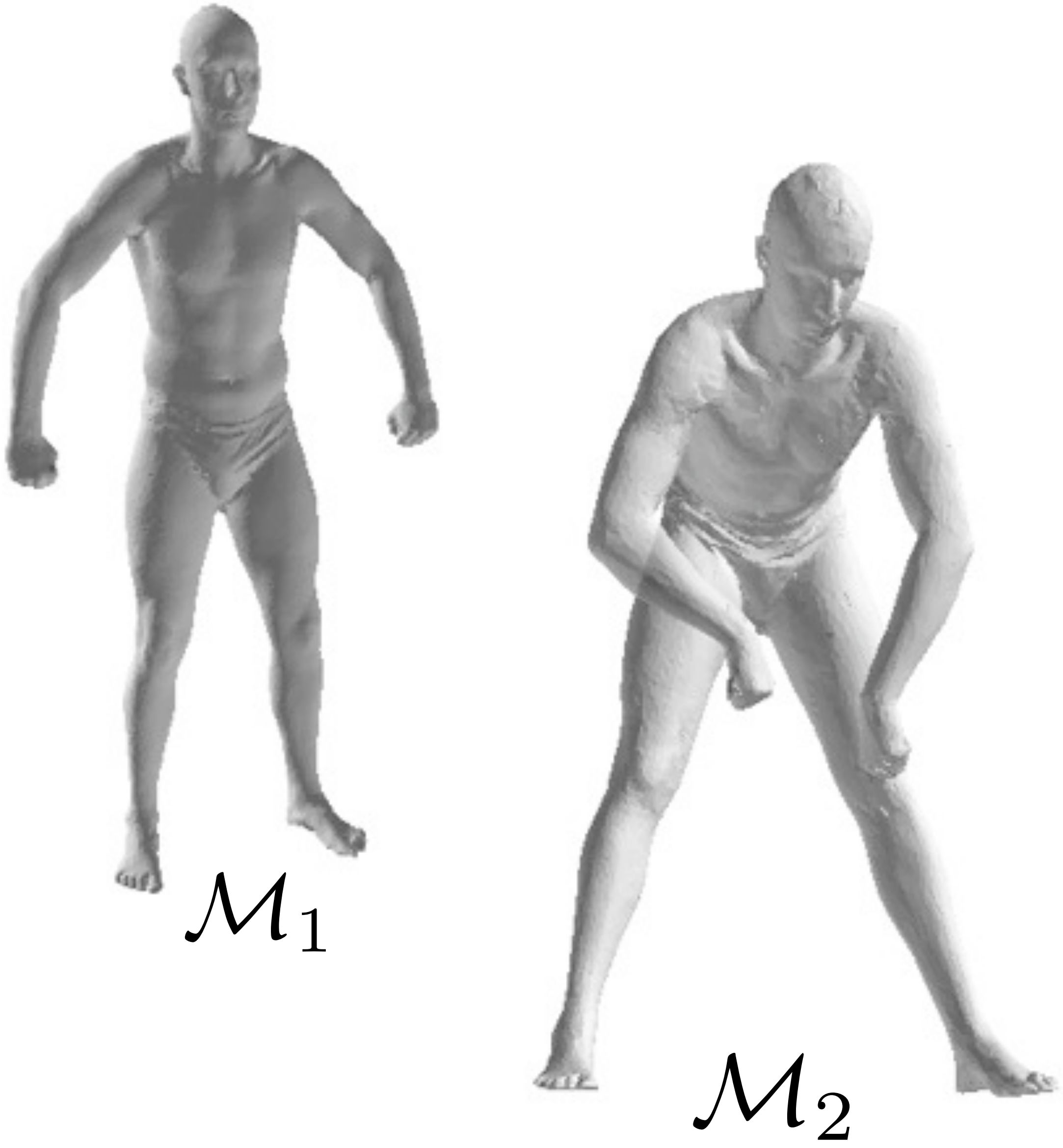
1. **Blended Intrinsic Maps**
2. Fuzzy Correspondences
3. Deformable Template

Complexity: O(N^2)

Goal

Input

- **A pair** of manifold surfaces
- Related by a non-uniform (i.e. non-isometric) deformation



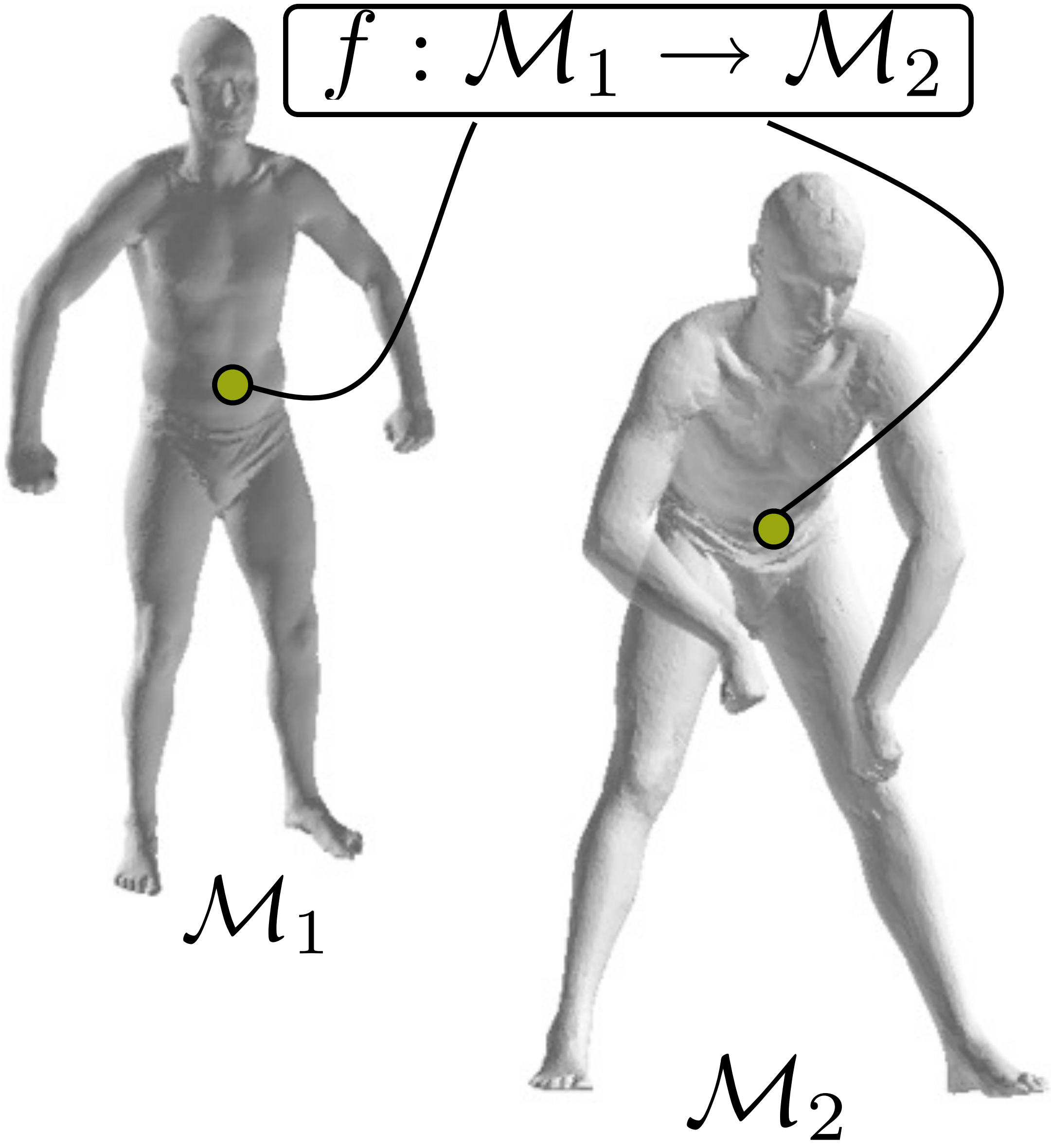
Goal

Input

- **A pair** of manifold surfaces
- Related by a non-uniform (i.e. non-isometric) deformation

Output

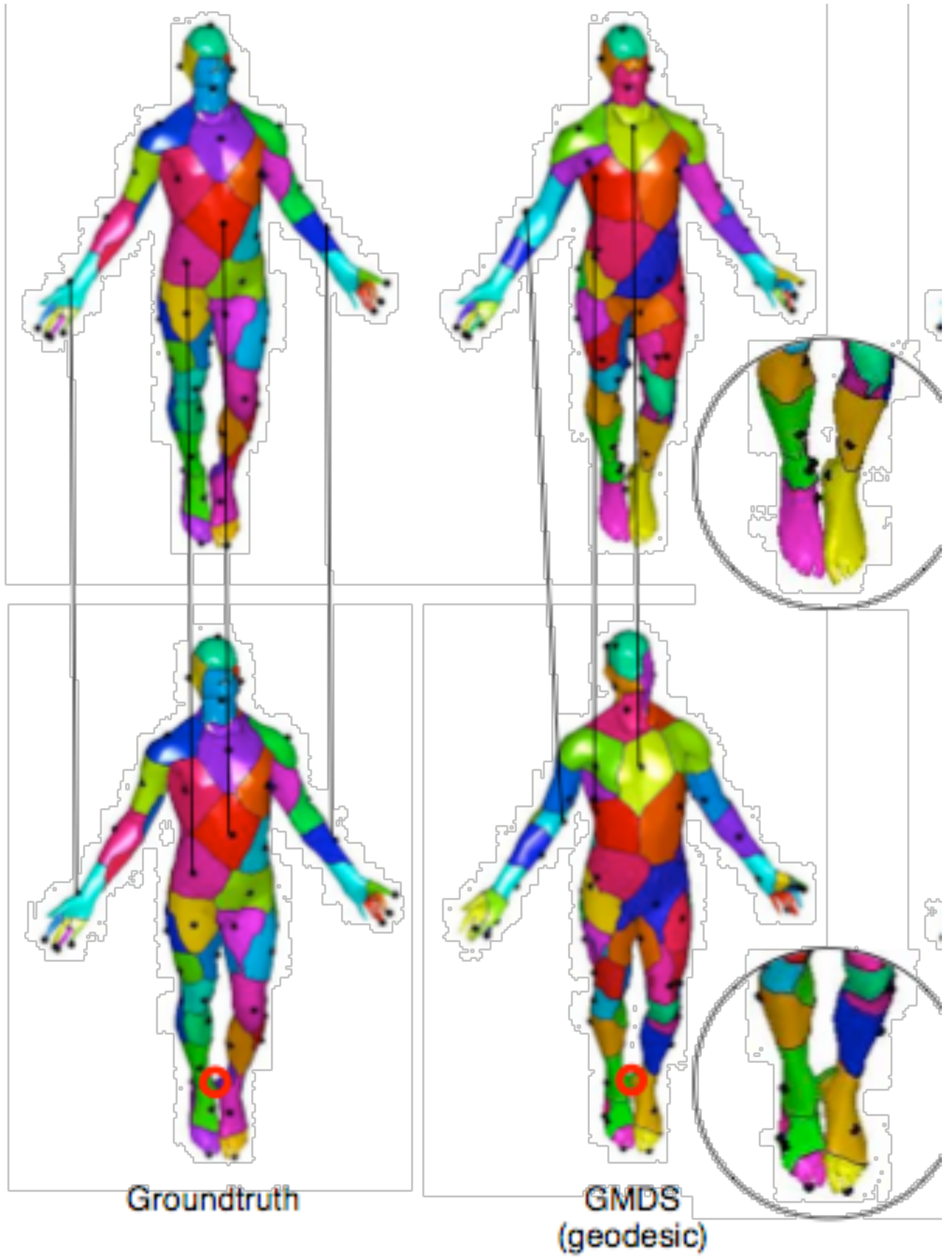
- A map defined at every point
- Smooth
- Low-distortion
- Aligns semantic features



Previous Work

Pairwise Correspondence: Intrinsic

- ➡ Gromov-Hausdorff
- o Möbius Transformations

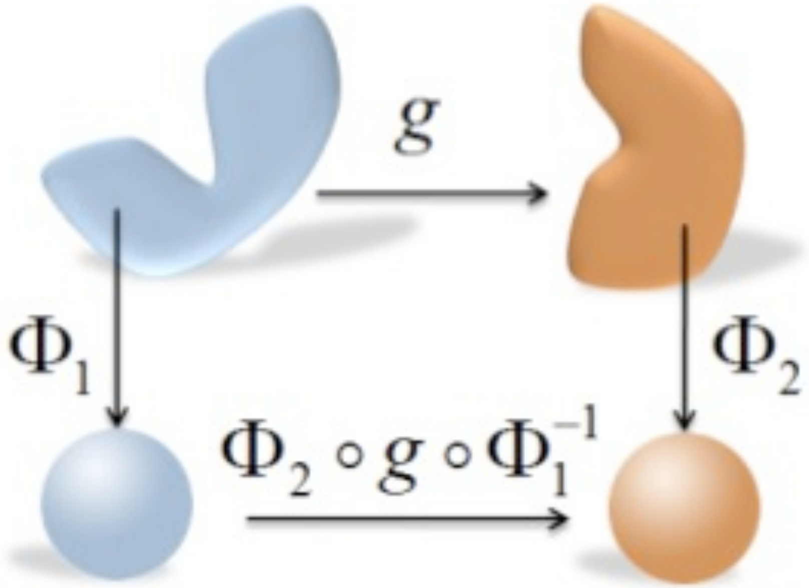


Bronstein et al., PNAS’ 06

Previous Work

Pairwise Correspondence: Intrinsic

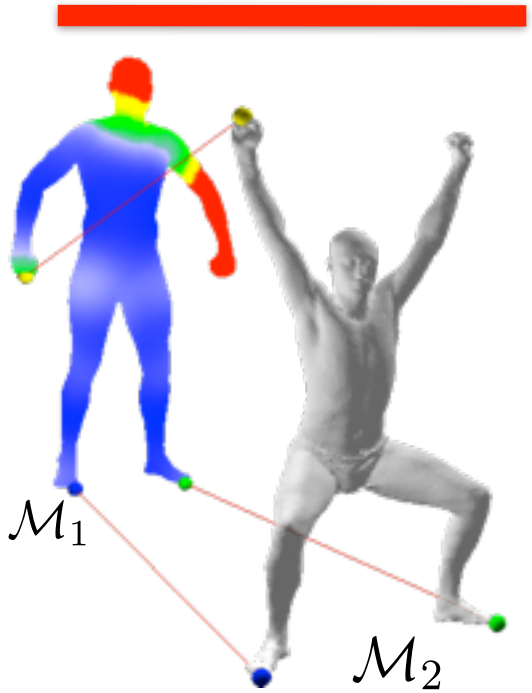
- ○ Gromov-Hausdorff
- ➡ Möbius Transformations



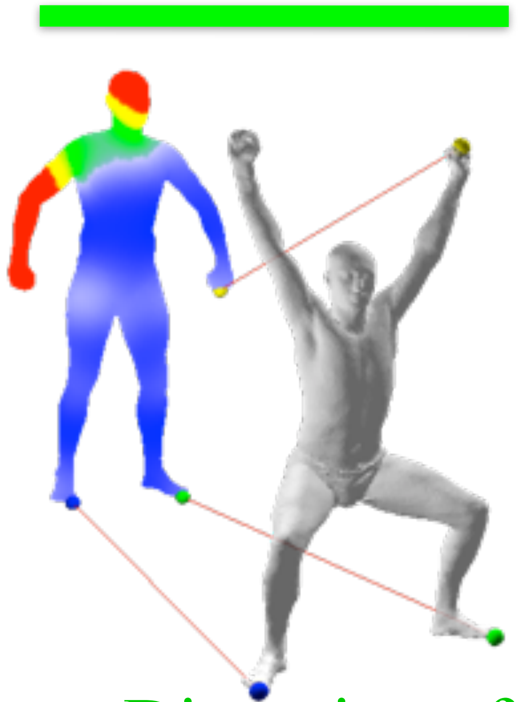
Lipman and Funkhouser, SIGGRAPH'09

Our Approach

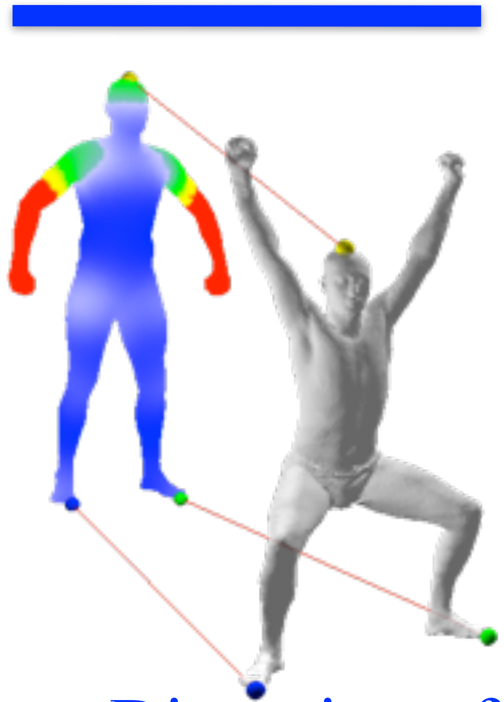
Weighted combination of locally-isometric maps



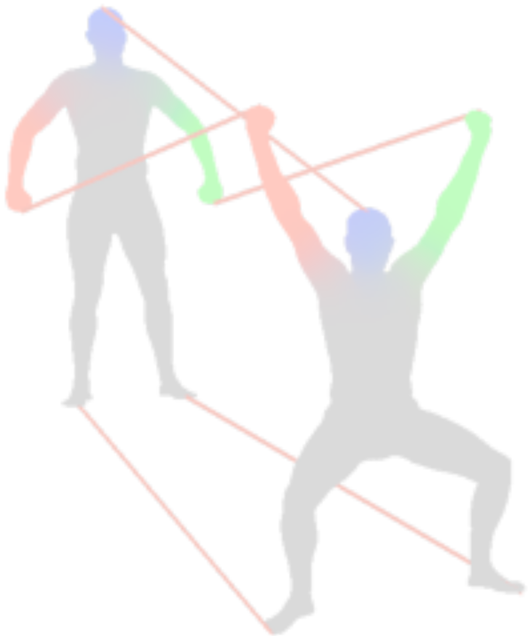
Distortion of m_1



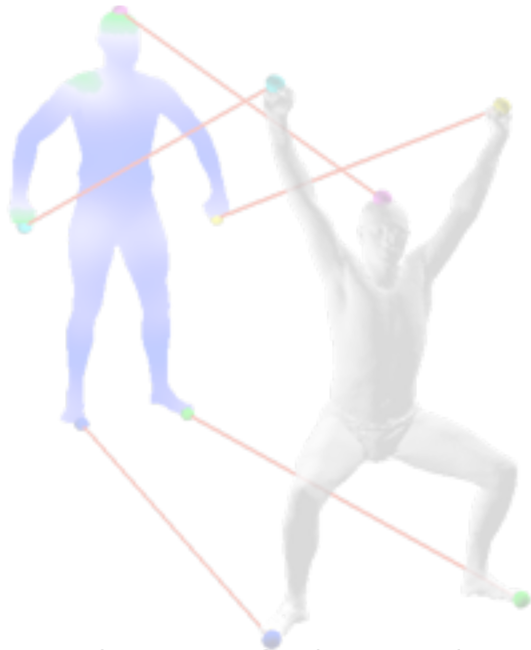
Distortion of m_2



Distortion of m_3



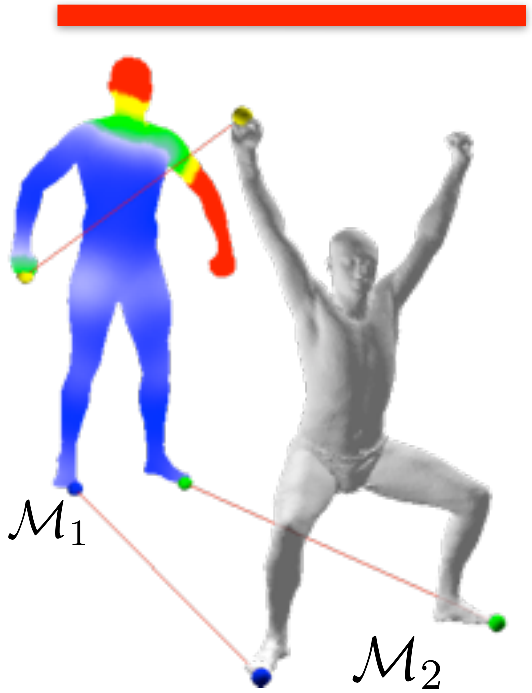
Blending Weights for m_1, m_2, and m_3



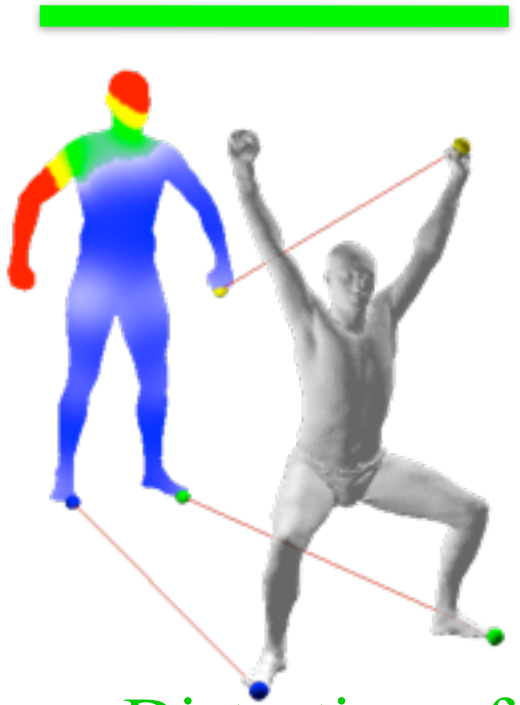
Distortion of the Blended Map

Our Approach

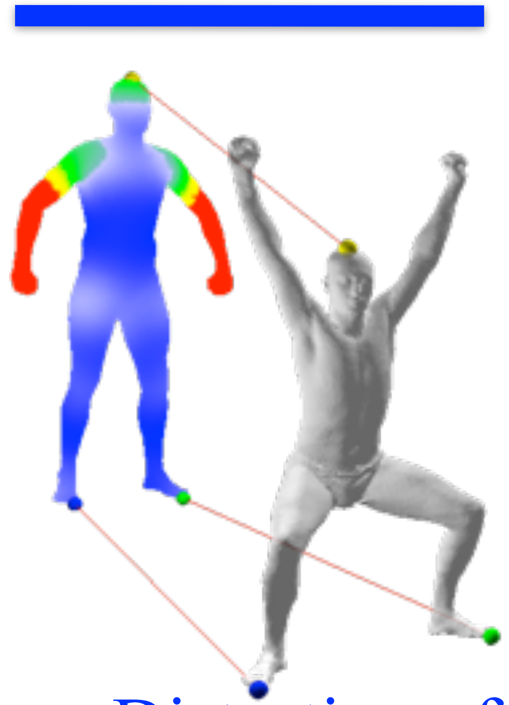
Weighted combination of locally-isometric maps



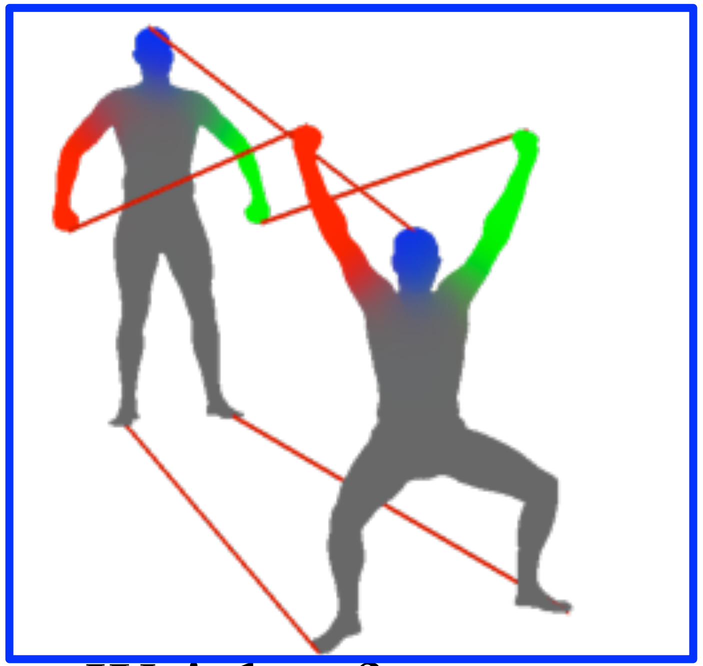
Distortion of m_1



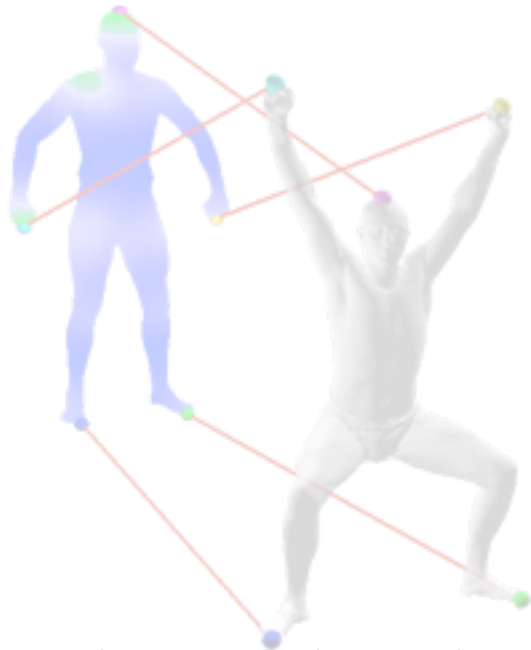
Distortion of m_2



Distortion of m_3



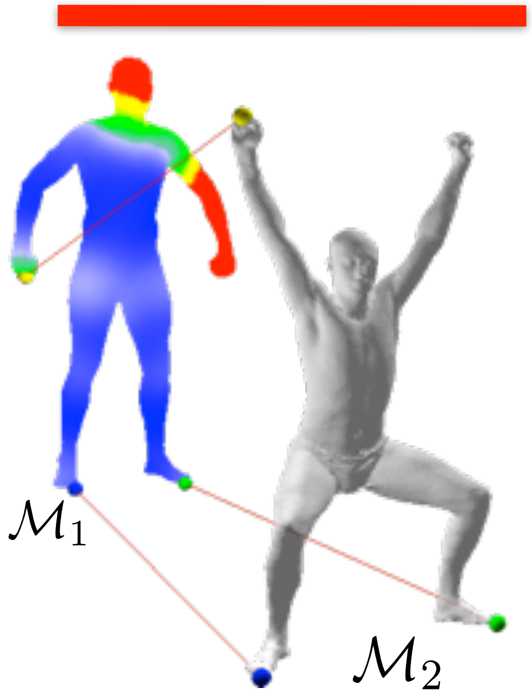
Blending Weights for m_1, m_2, and m_3



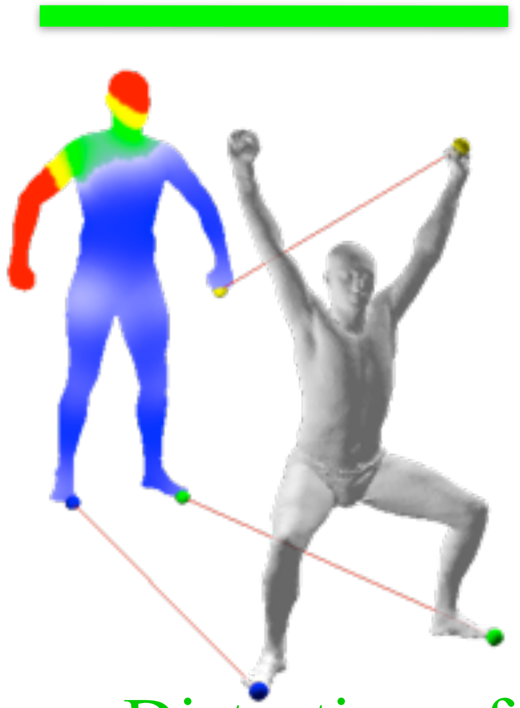
Distortion of the Blended Map

Our Approach

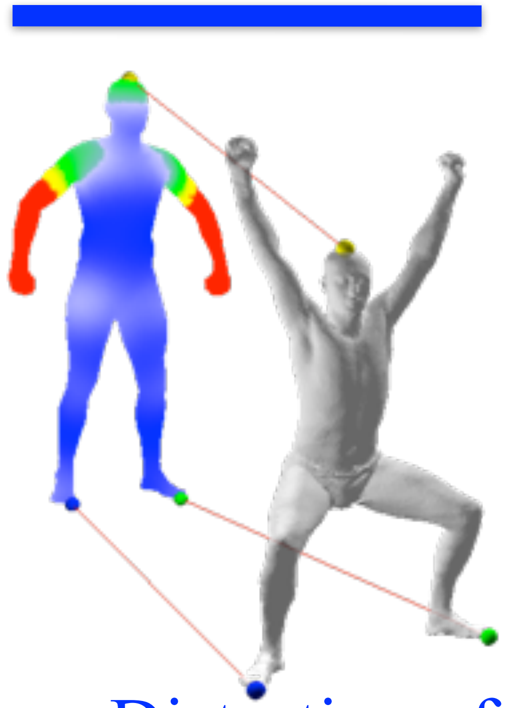
Weighted combination of locally-isometric maps



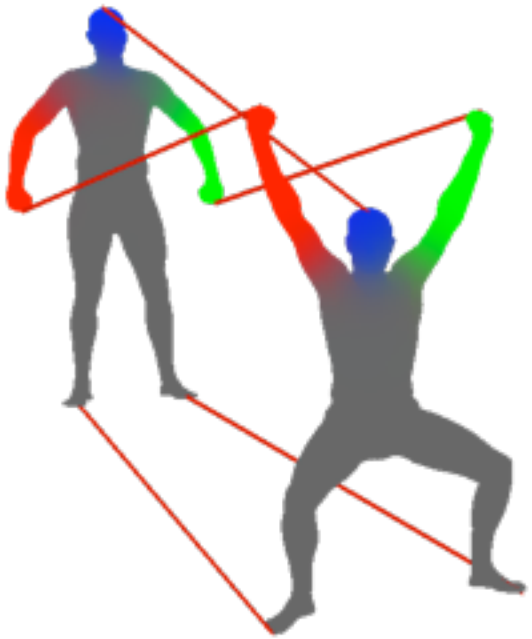
Distortion of m_1



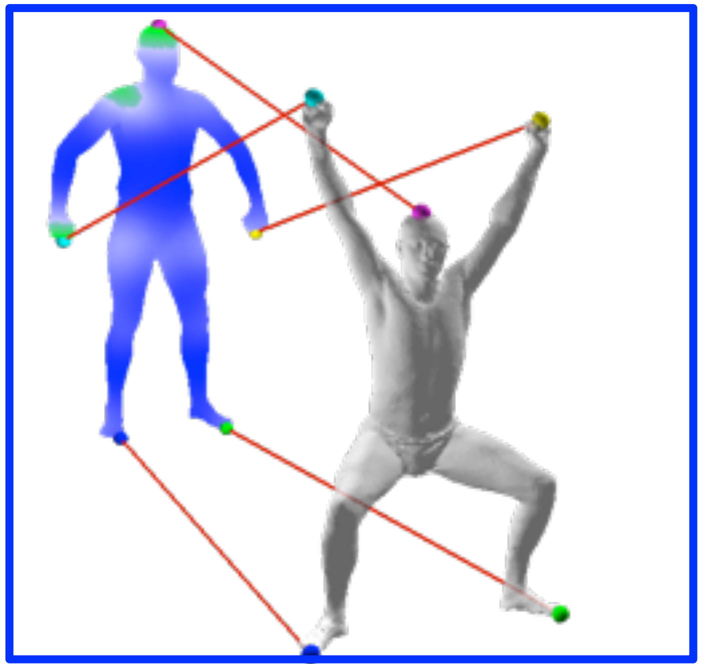
Distortion of m_2



Distortion of m_3

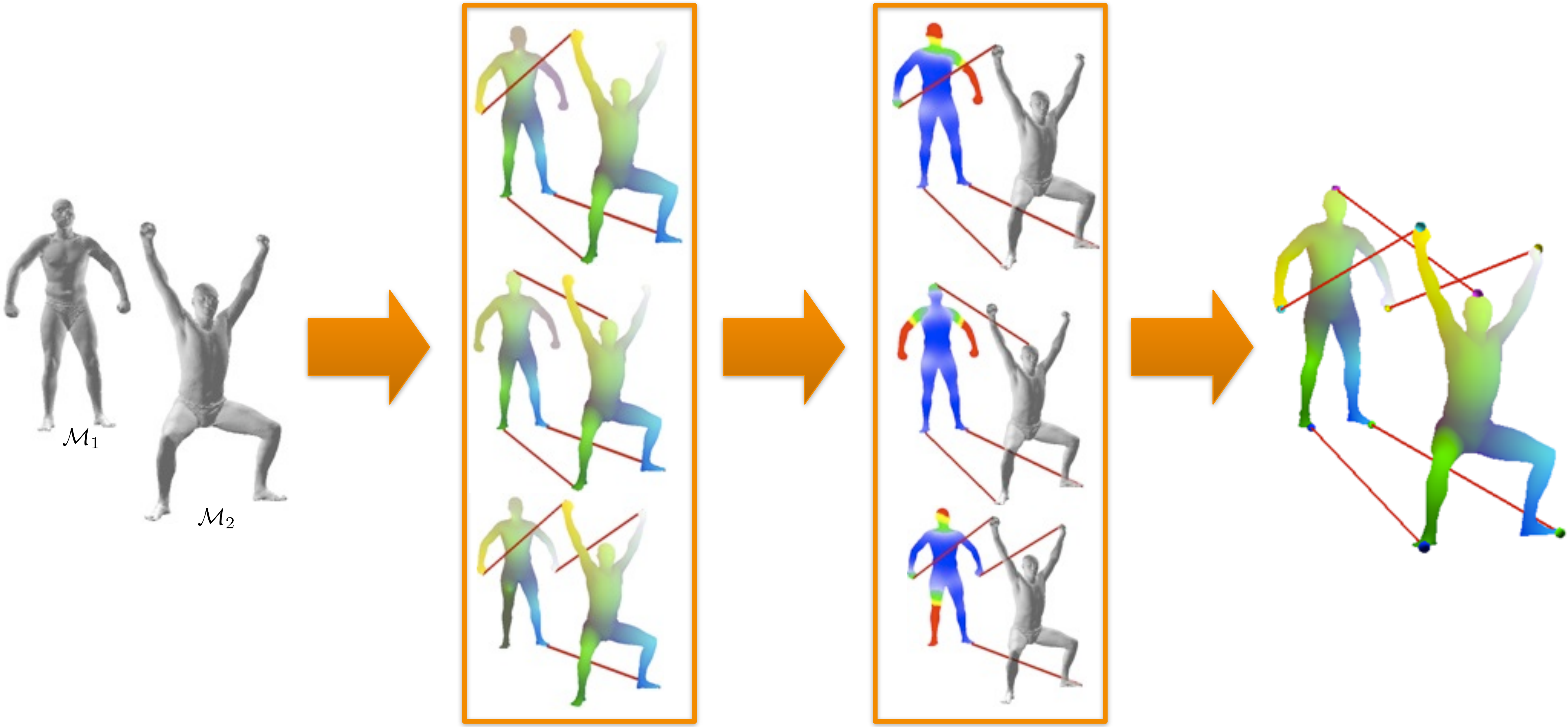


Blending Weights for m_1, m_2, and m_3



Distortion of the Blended Map

The Computational Pipeline

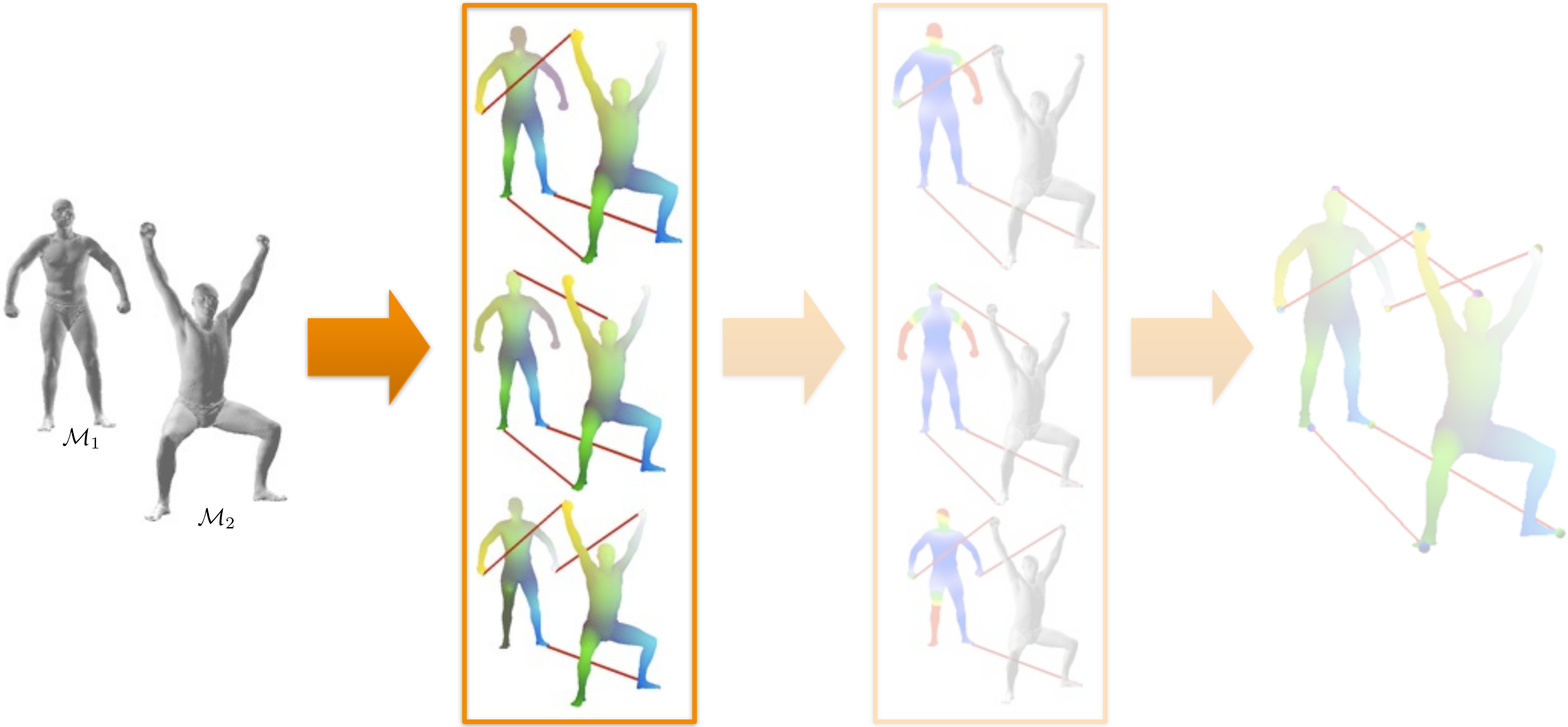


Generate consistent set of maps

Find blending weights

Blend maps

The Computational Pipeline



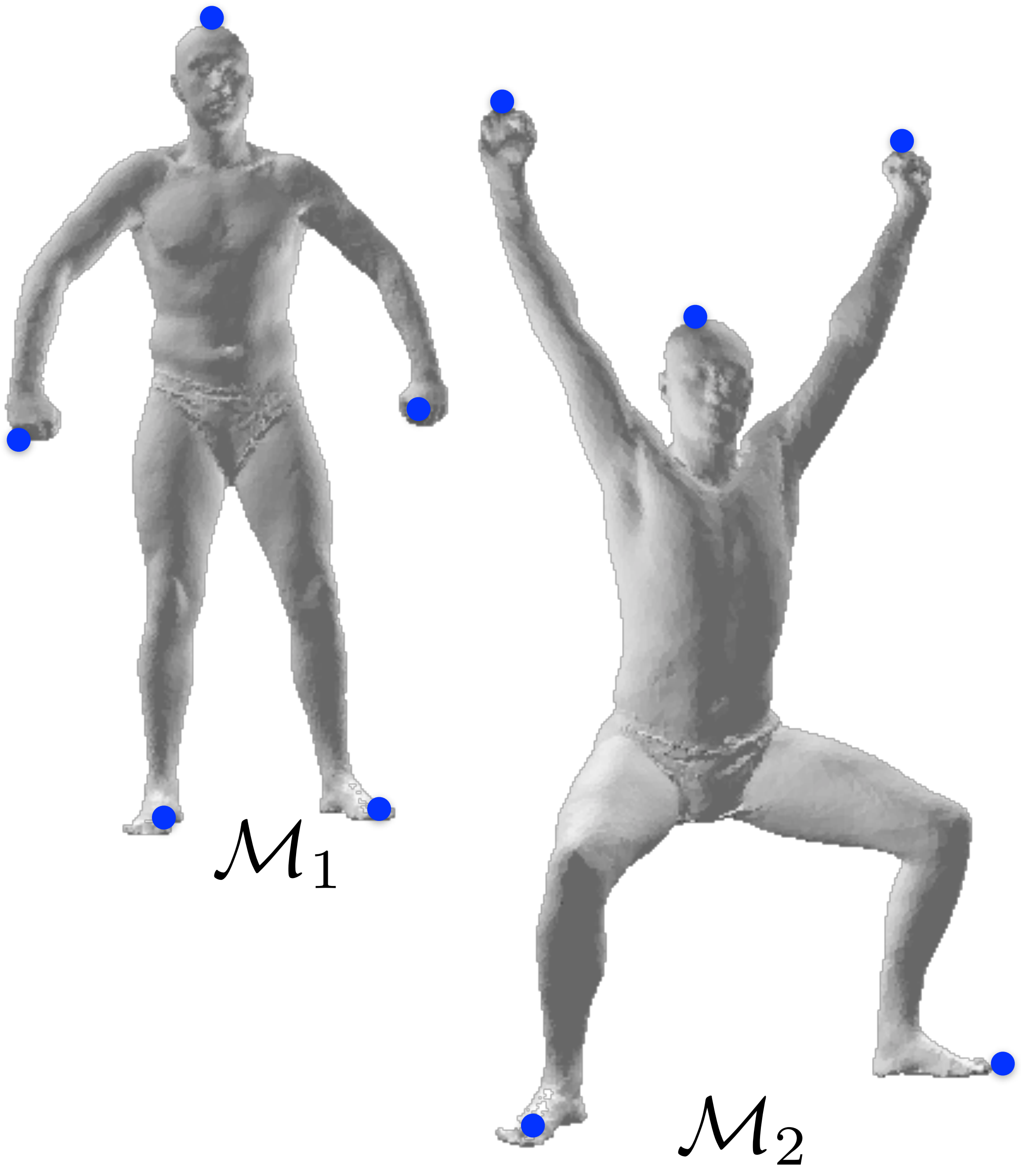
Generate consistent set of maps

Find blending weights

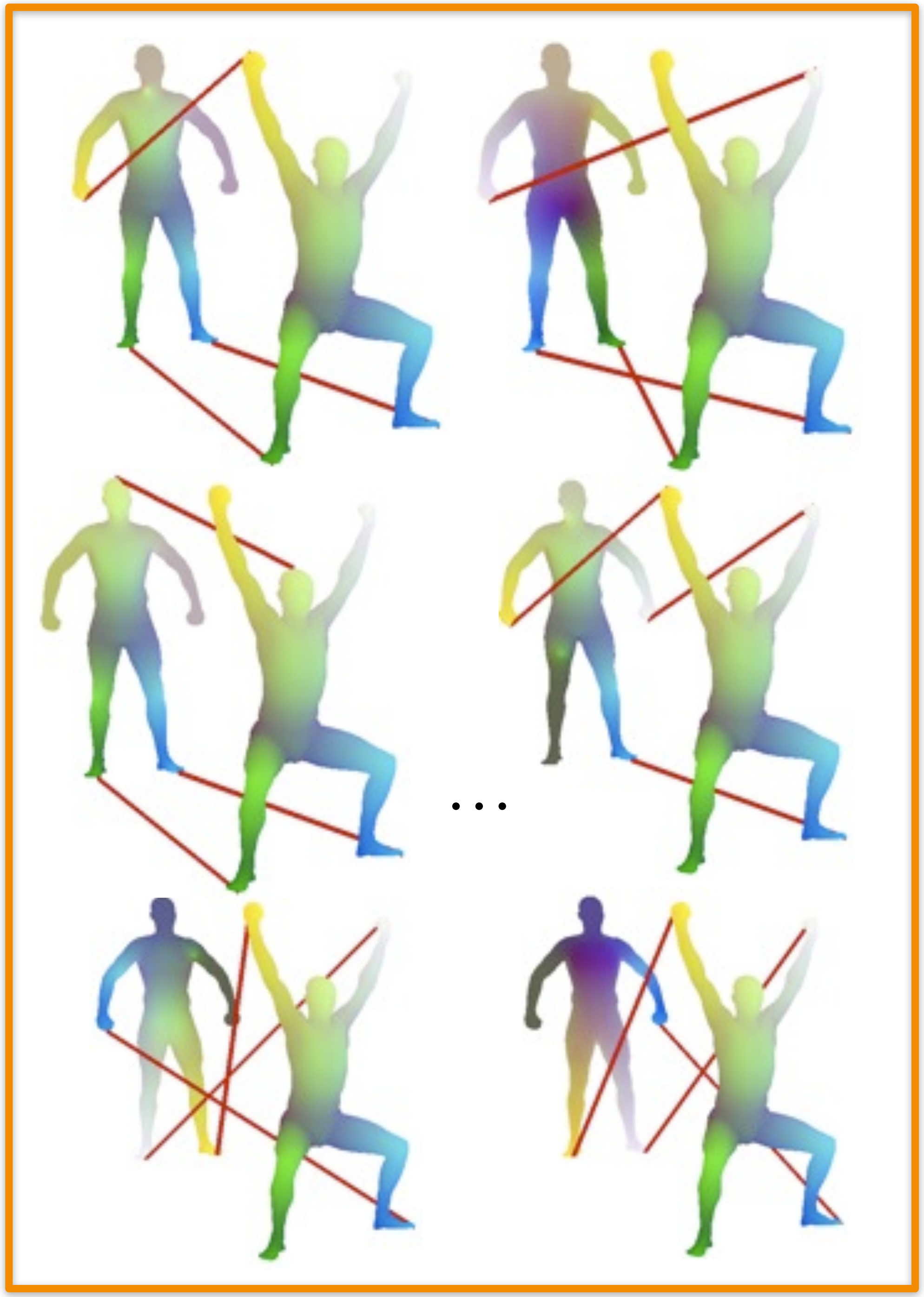
Blend maps

Generating Consistent Maps

Generate a set of candidate conformal maps by enumerating triplets of feature points

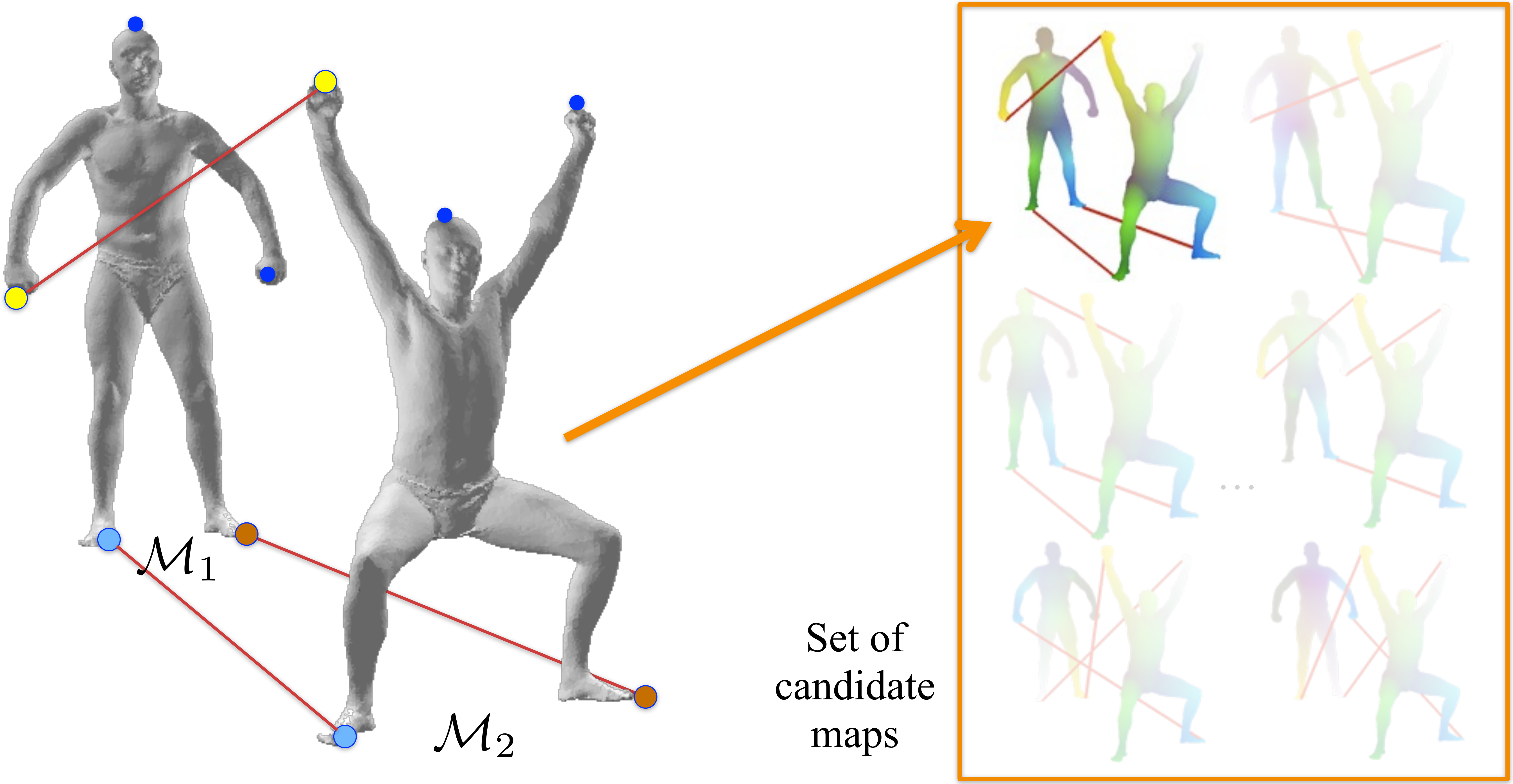


Set of candidate maps



Generating Consistent Maps

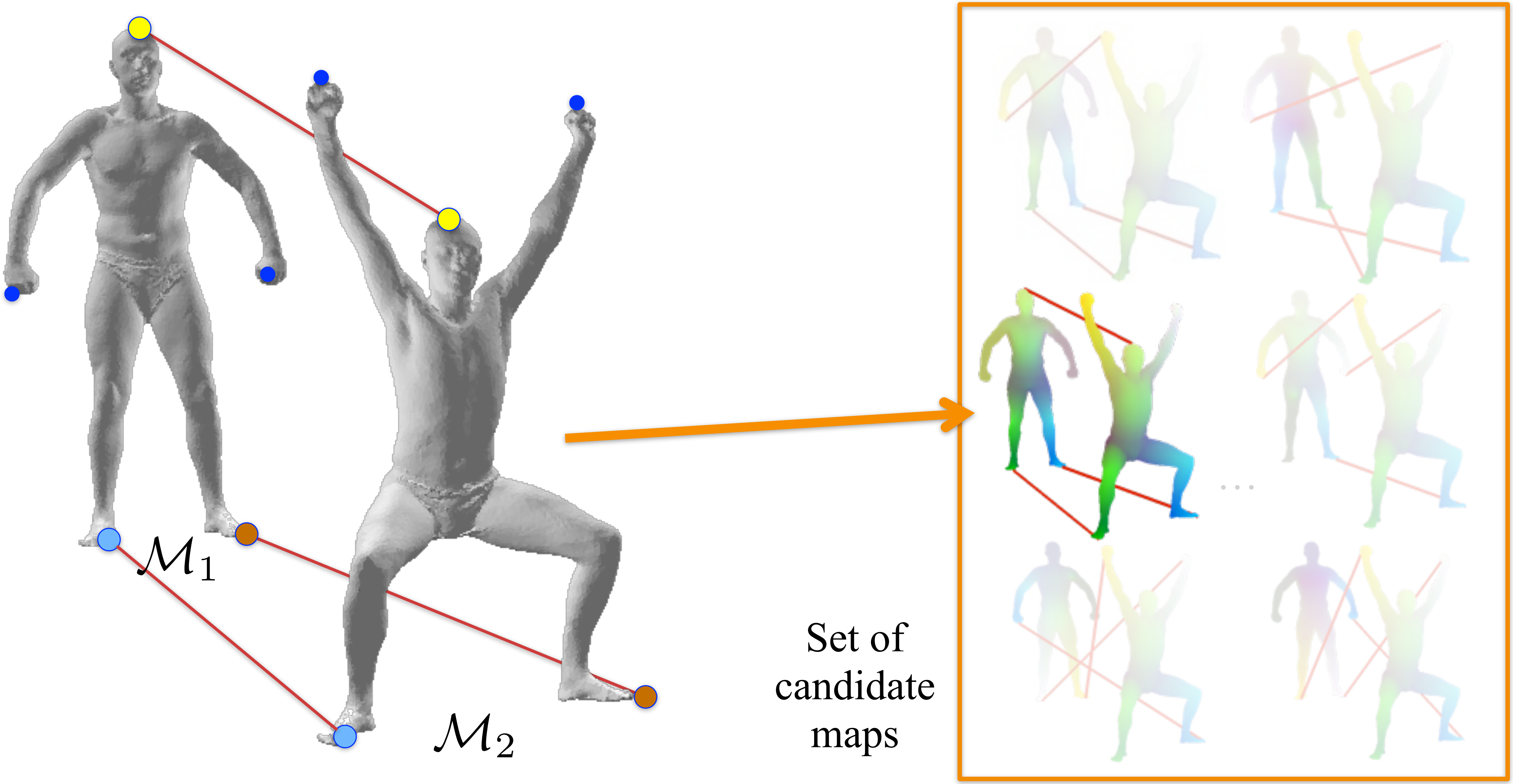
Generate a set of candidate conformal maps by enumerating triplets of feature points



Set of candidate maps

Generating Consistent Maps

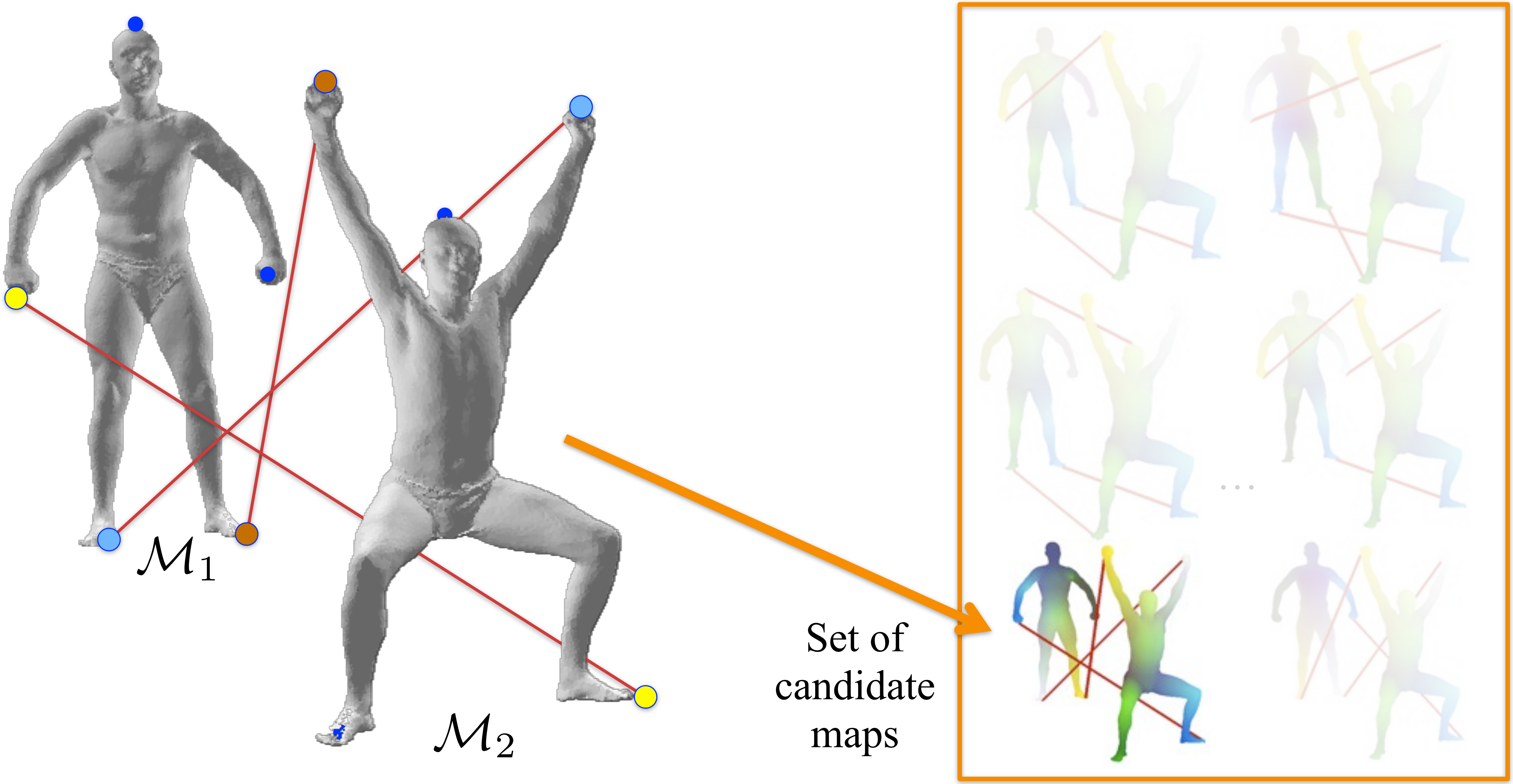
Generate a set of candidate conformal maps by enumerating triplets of feature points



Set of candidate maps

Generating Consistent Maps

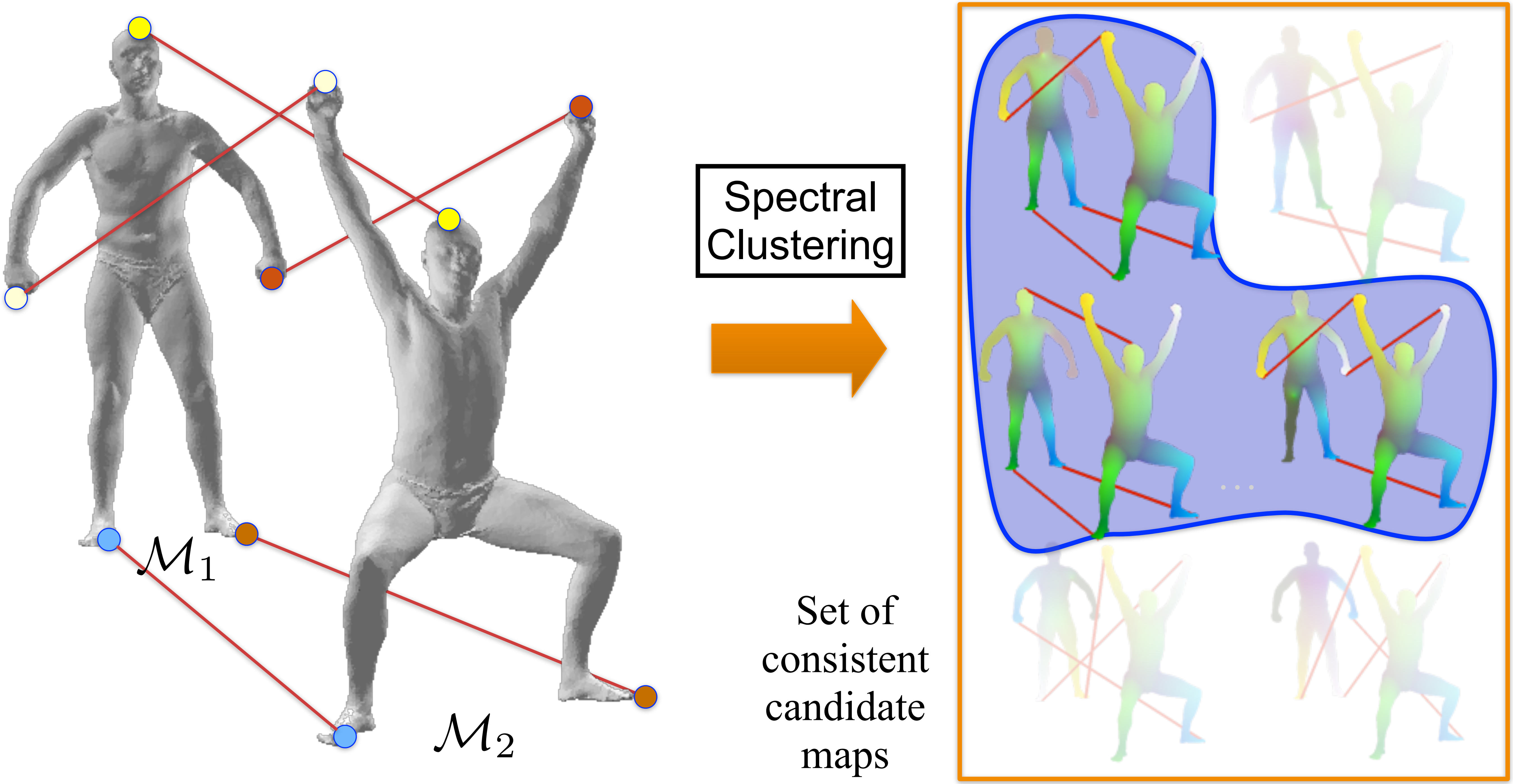
Generate a set of candidate conformal maps by enumerating triplets of feature points



Set of candidate maps

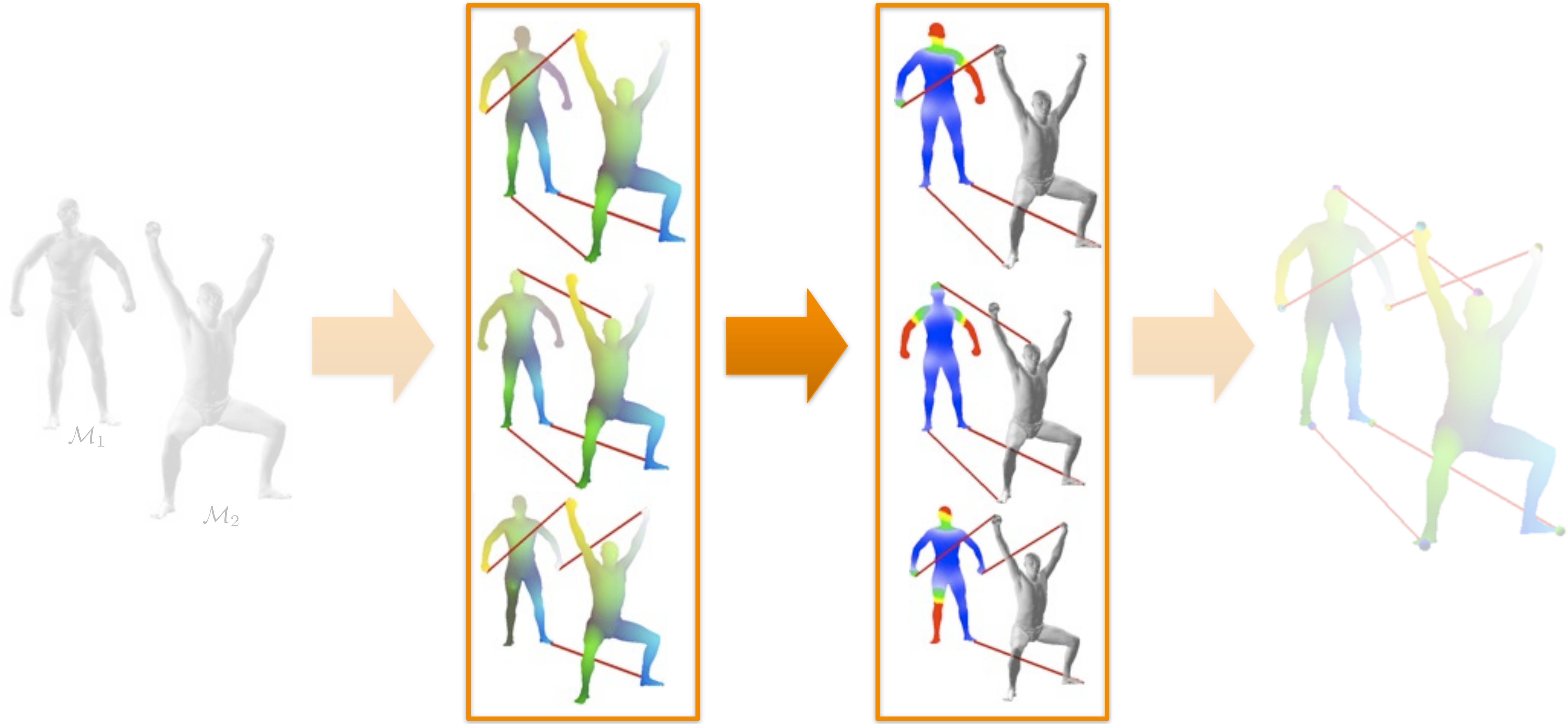
Generating Consistent Maps

Find consistent set of candidate maps



Set of consistent candidate maps

The Computational Pipeline



Generate consistent set of maps

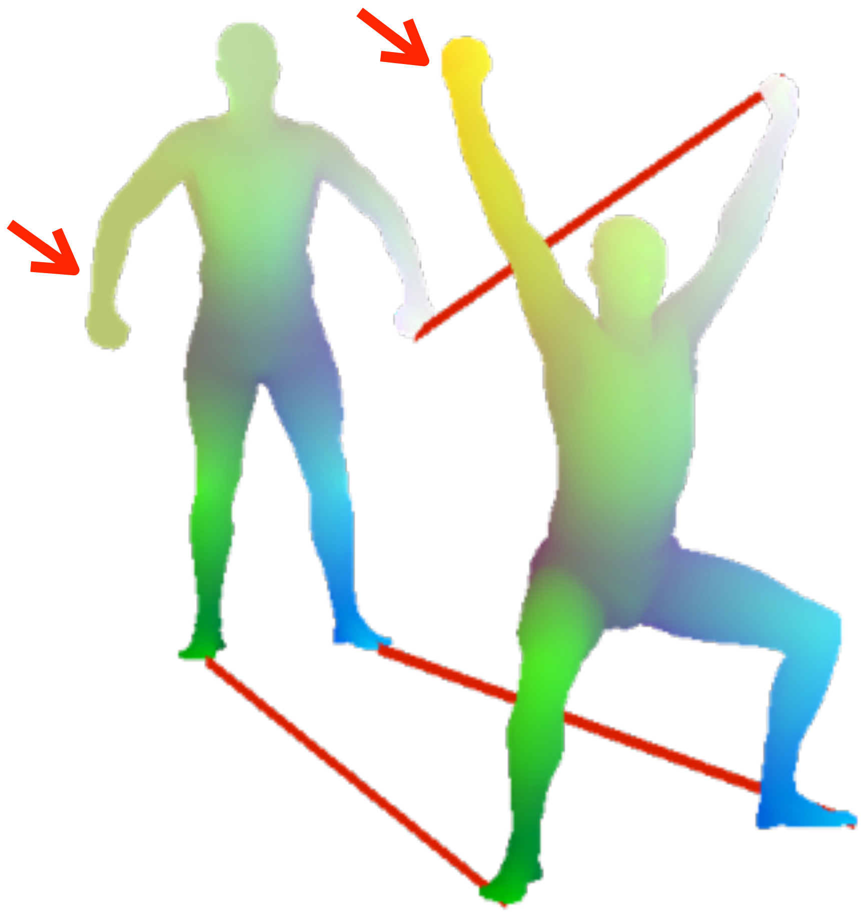
Find blending weights

Blend maps

Finding Blending Weights

For every point p

- Compute a weight of each map m_i at p



Candidate Map

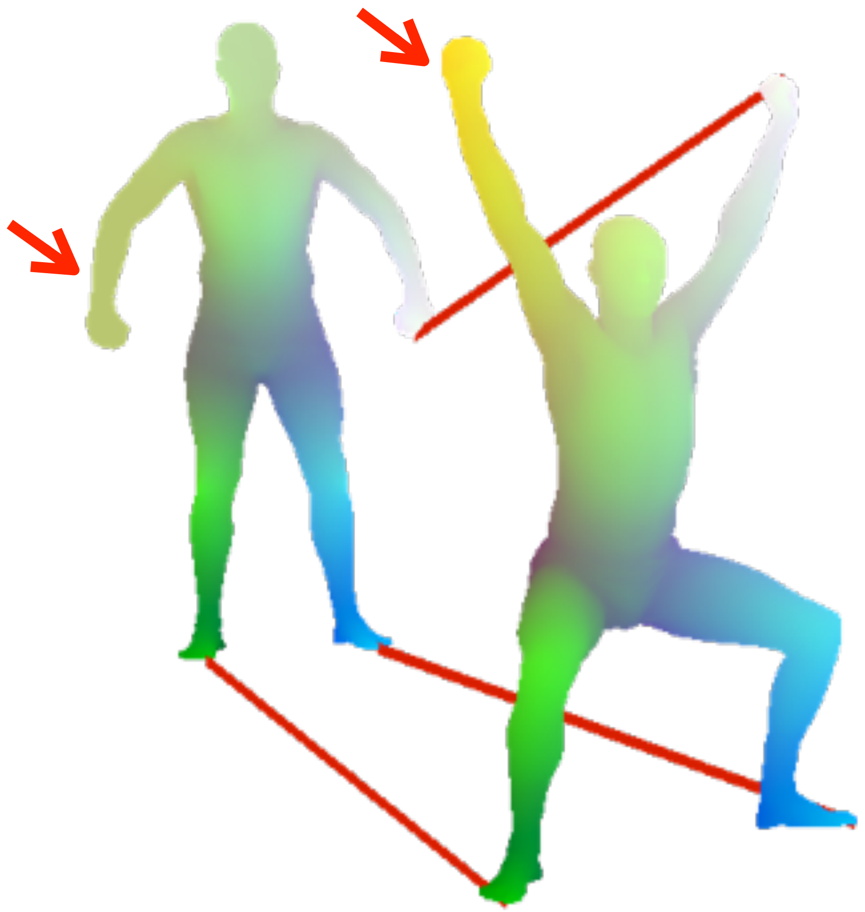
Finding Blending Weights

For every point p

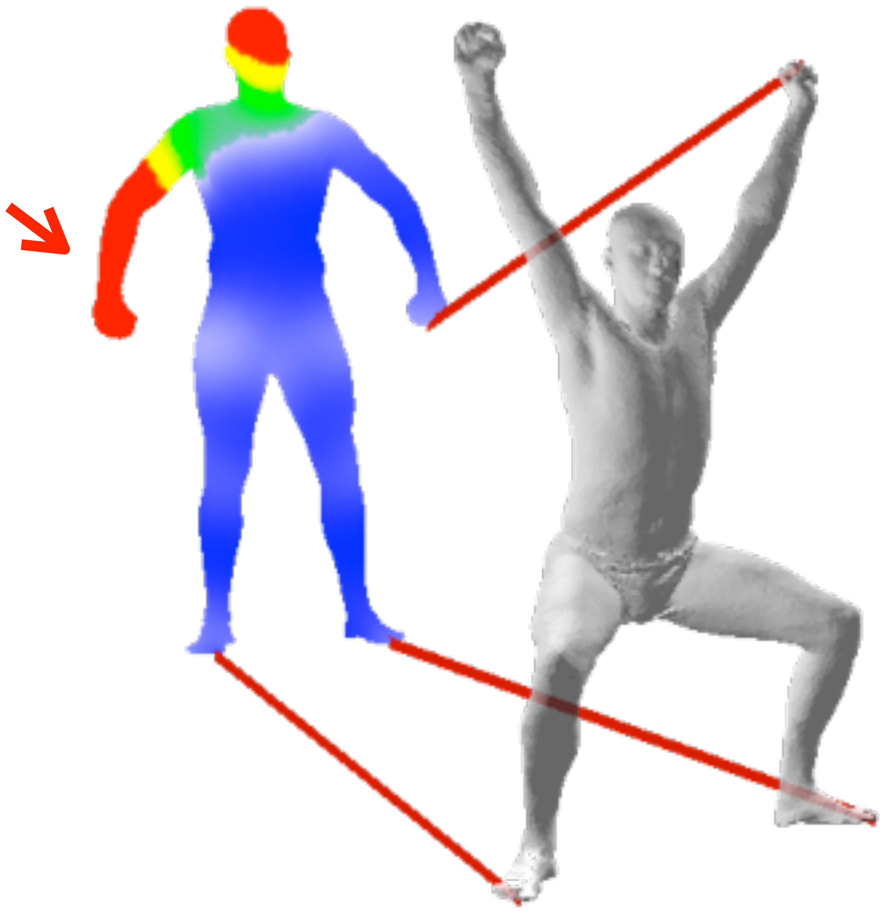
- Compute a weight of each map m_i at p

We model the weight with deviation from isometry

- Area distortion for conformal maps

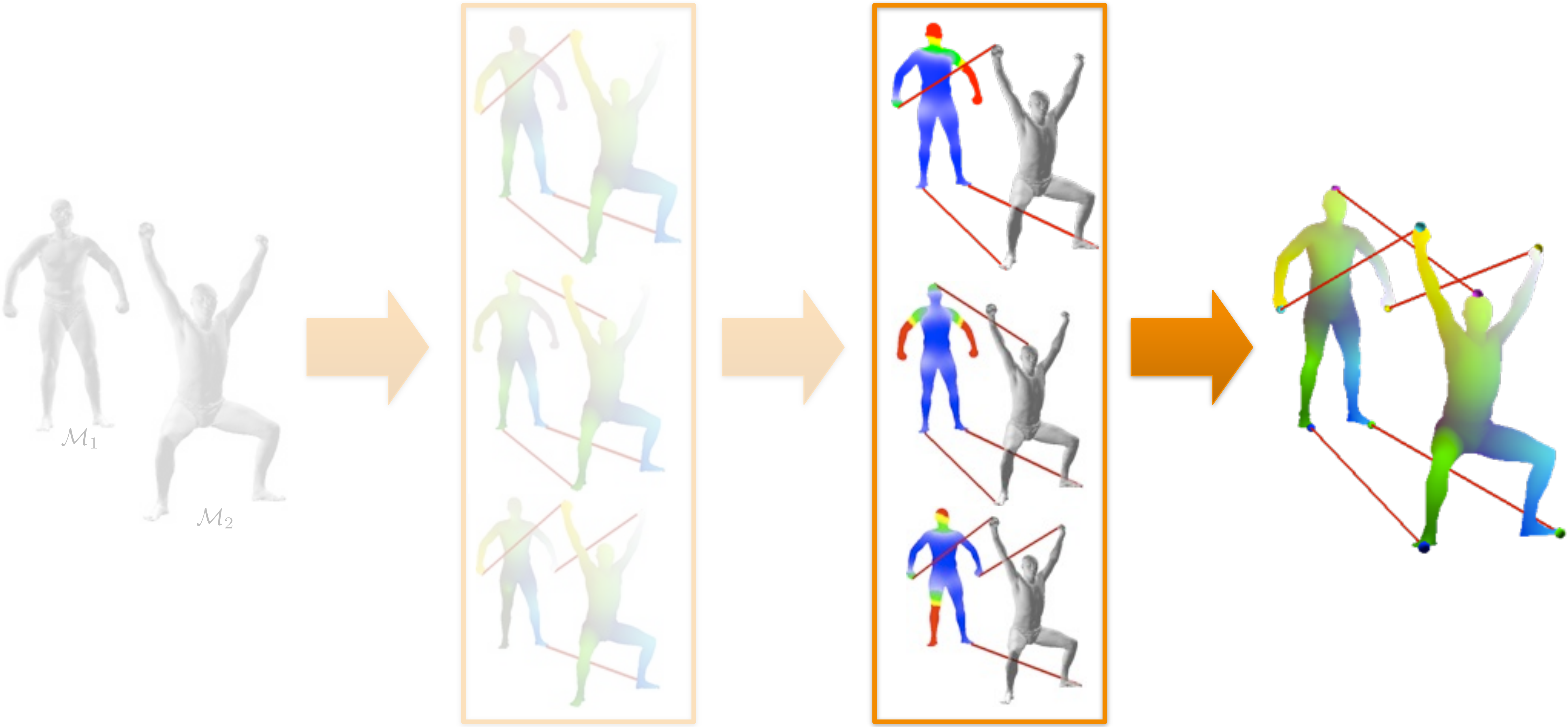


Candidate Map



Blending Weight $c_i(p)$

The Computational Pipeline



Generate consistent set of maps

Find blending weights

Blend maps

Blending Maps

Input for each point p:

- An image $m_i(p)$ after applying each map m_i
- A blending weight for each map

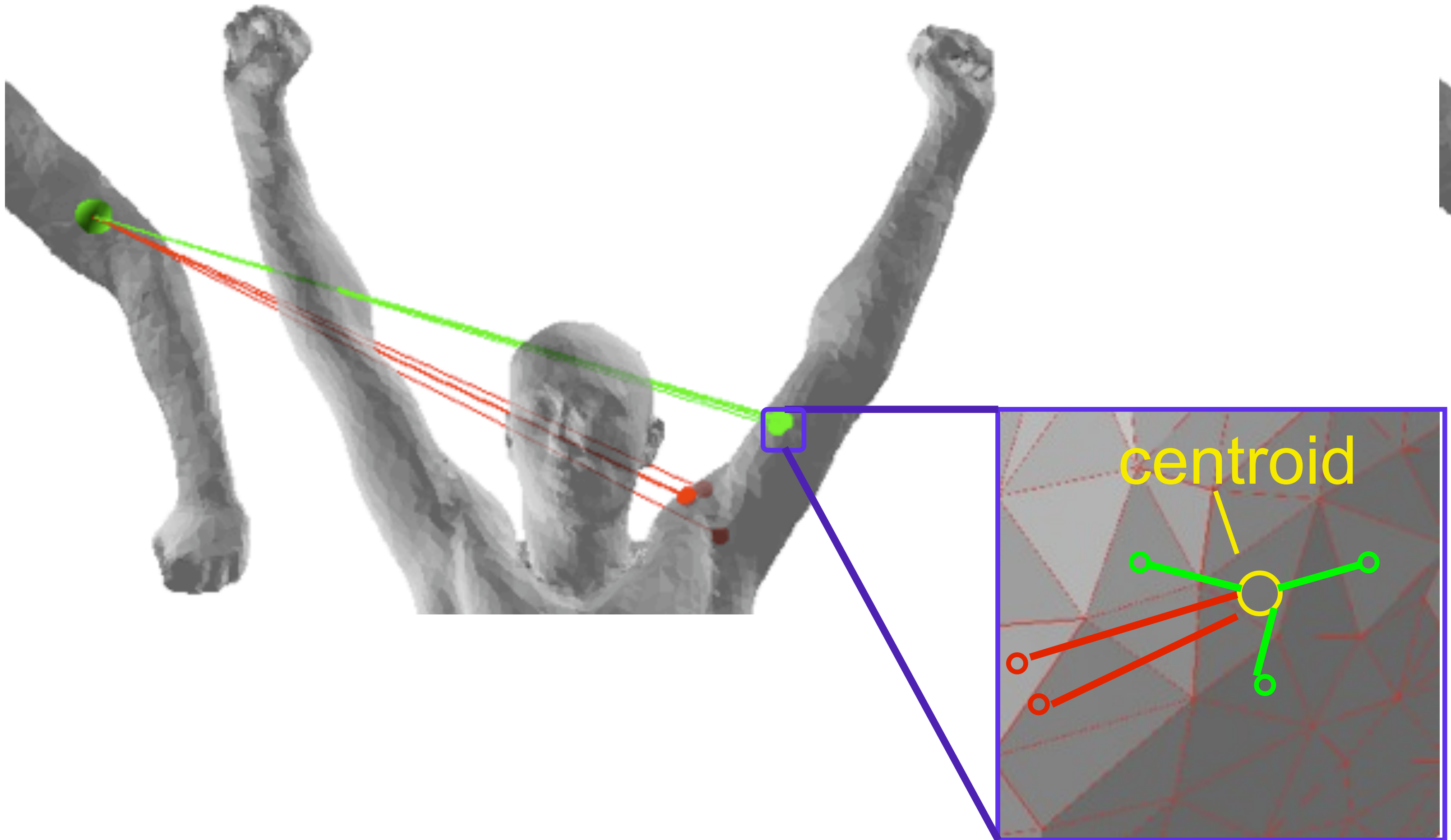
Blending Maps

Input for each point p:

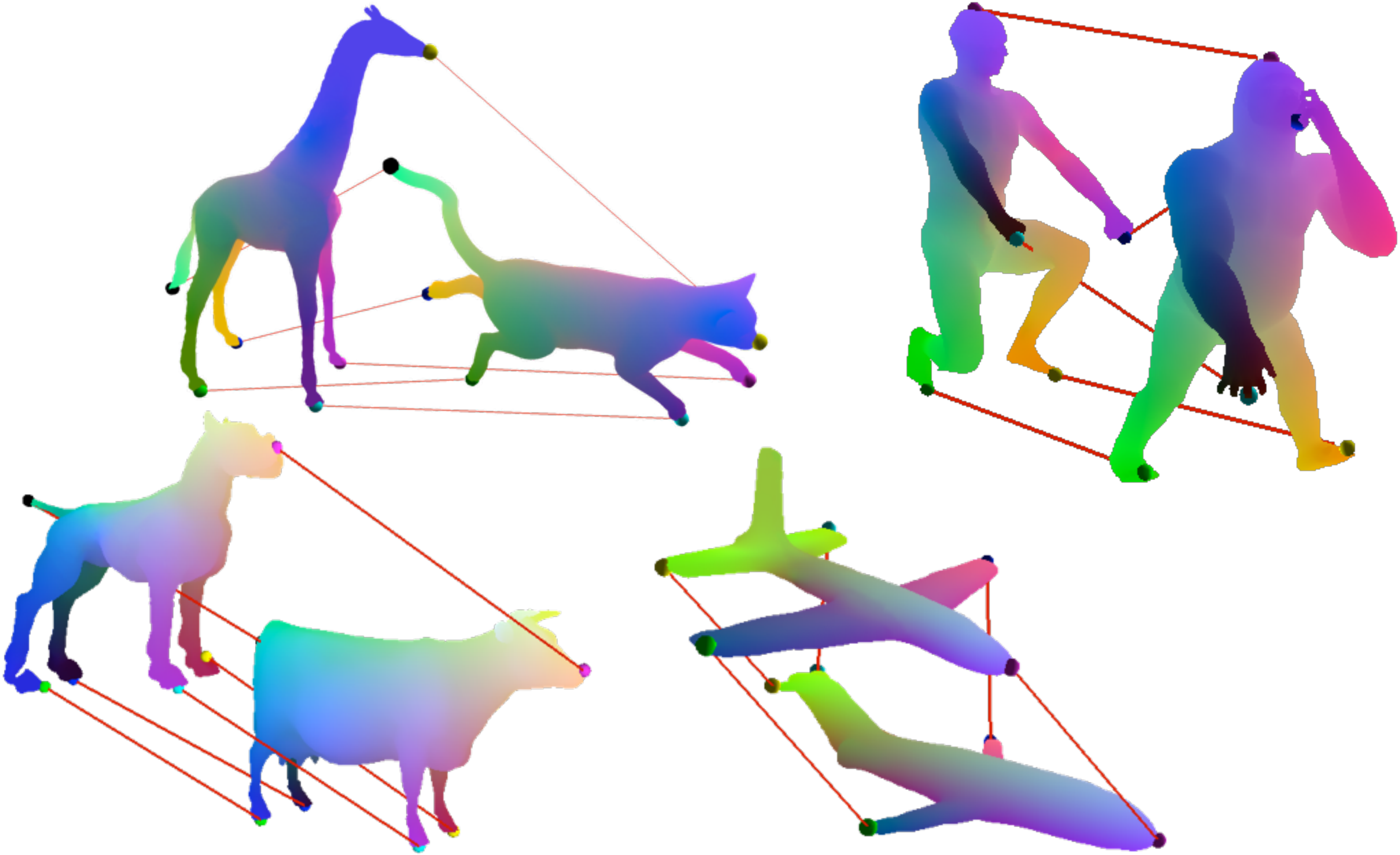
- An image $m_i(p)$ after applying each map m_i
- A blending weight for each map

Output for each point:

- Weighted geodesic centroid of $\{ m_i(p) \}$

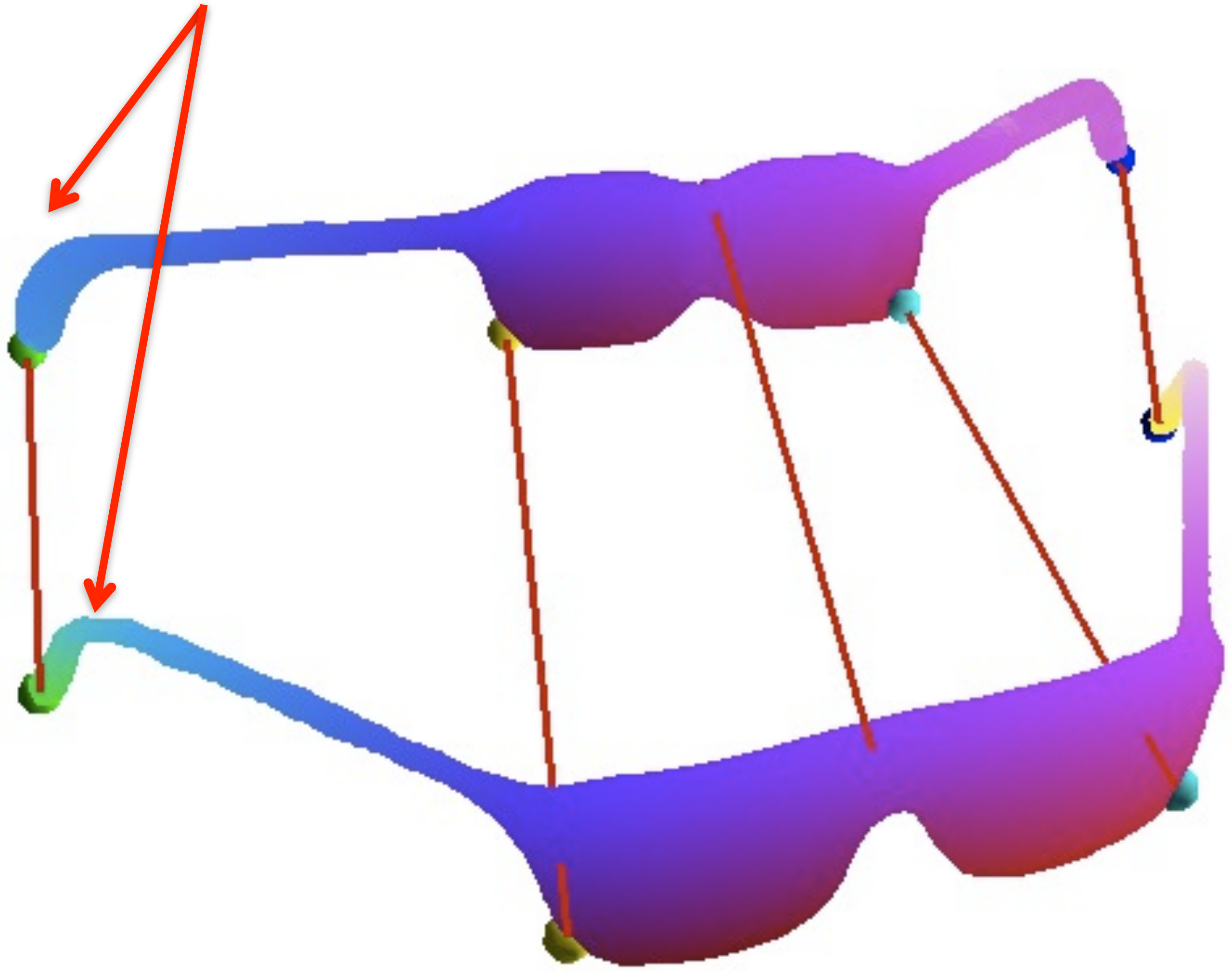


Results

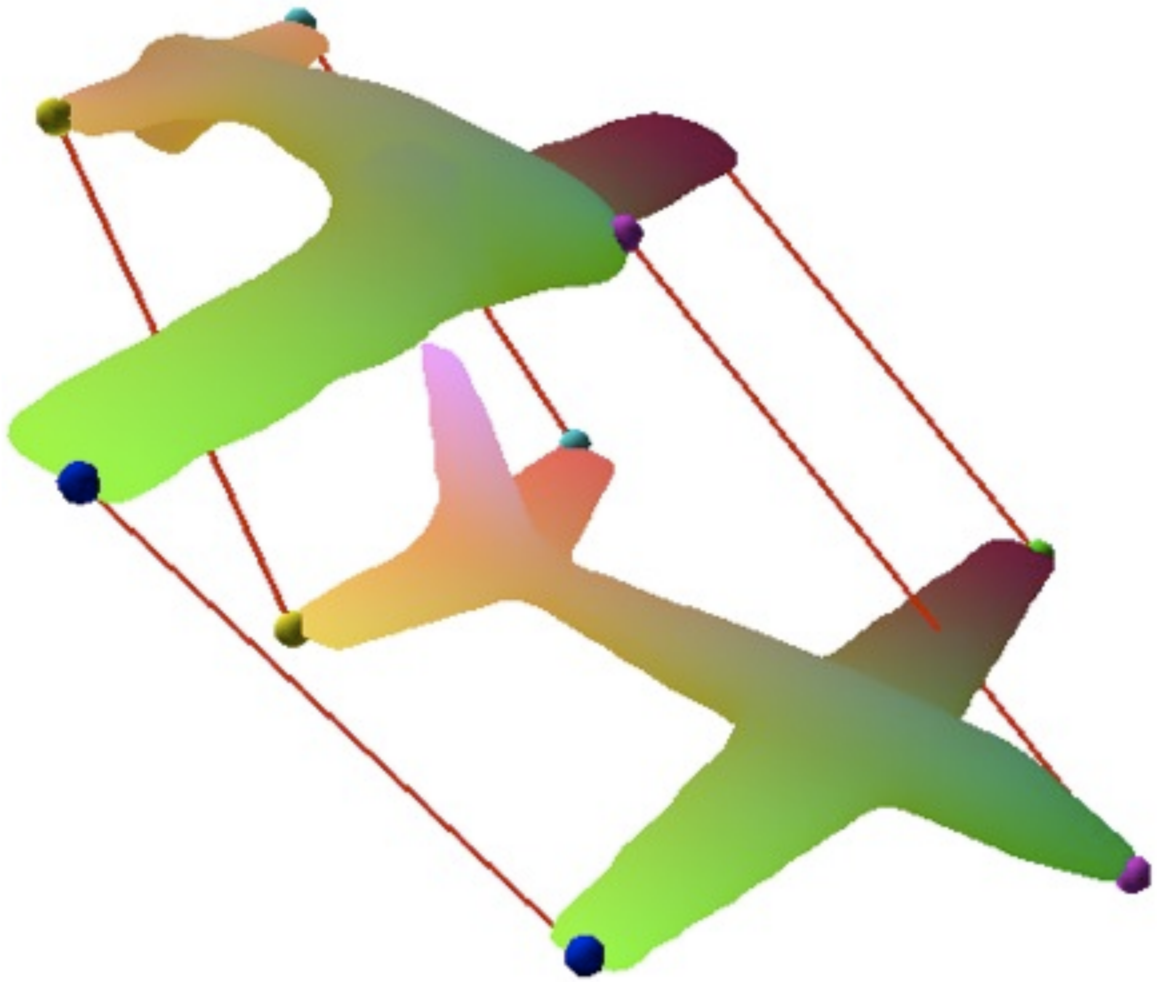


Failures

Stretched

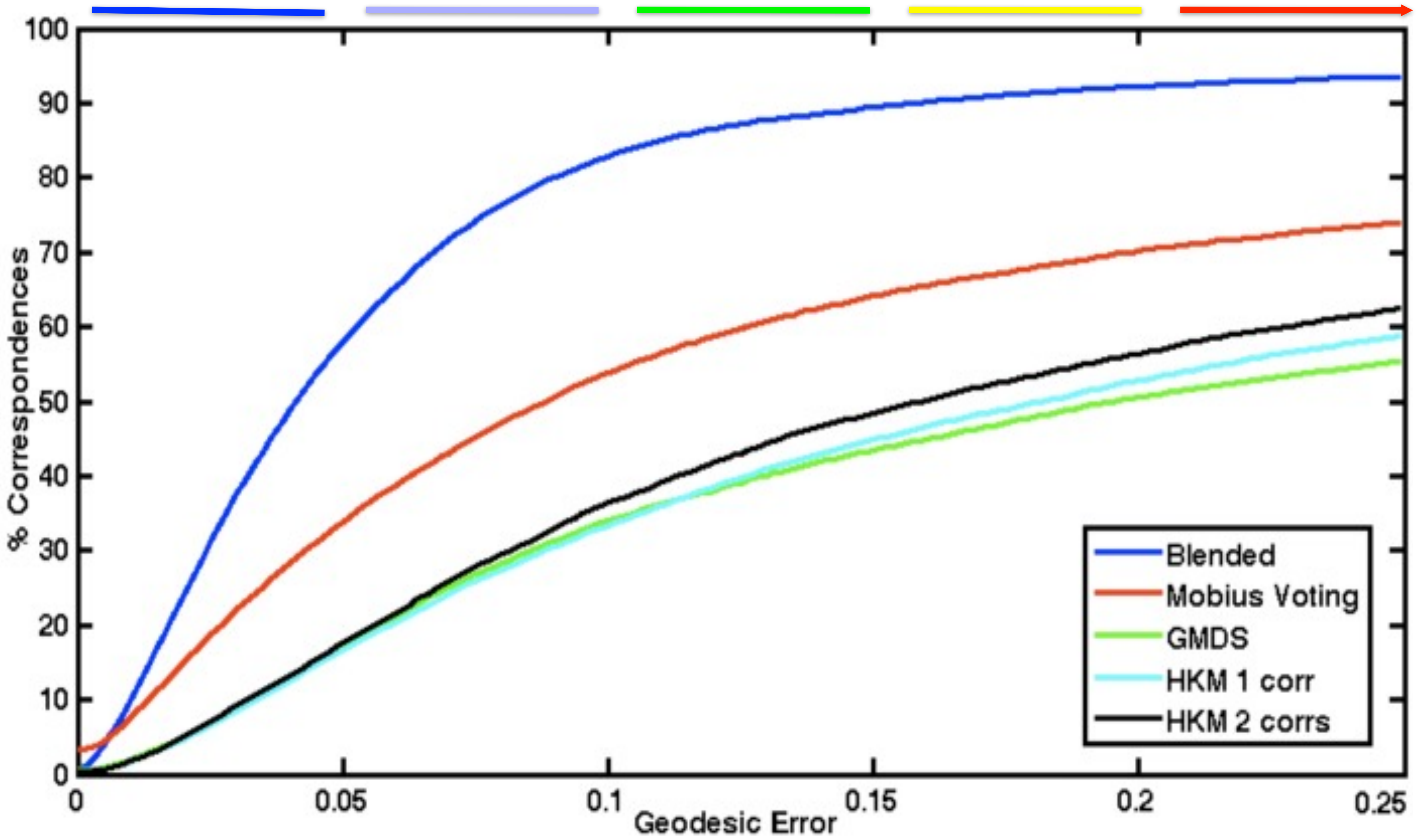


Symmetric flip



Comparison

$0 \leq d < 0.05$ $0.05 \leq d < 0.1$ $0.1 \leq d < 0.15$ $0.15 \leq d < 0.2$ $0.2 \leq d < \infty$

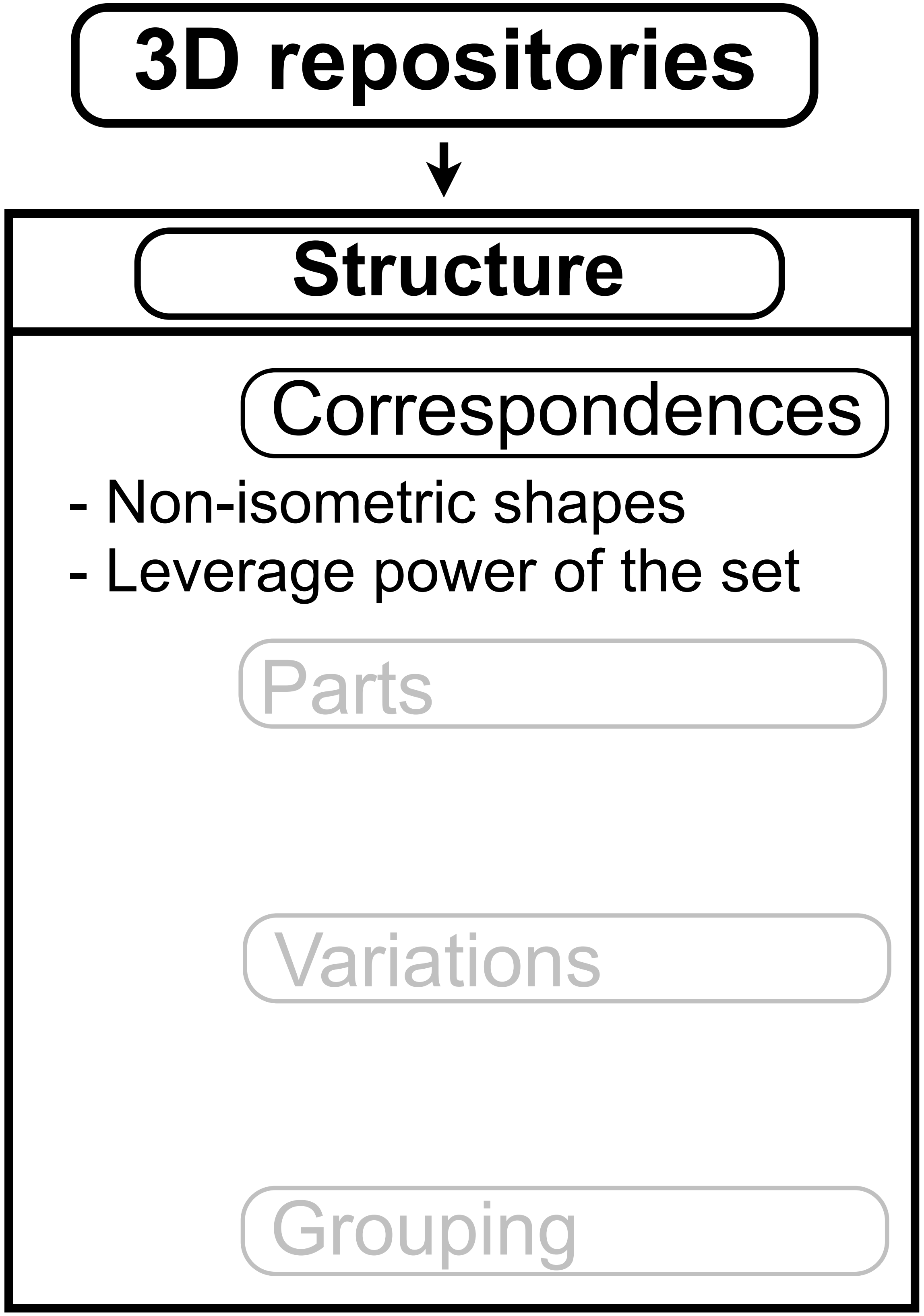


Summary

Blend locally-isometric maps

- Robust to large non-uniform deformations
- Efficient to compute
- Outperforms other methods on benchmark

Goal



3D repositories

Structure

Correspondences

- Non-isometric shapes
- Leverage power of the set

Parts

Variations

Grouping

1. Blended Intrinsic Maps

2. Fuzzy Correspondences

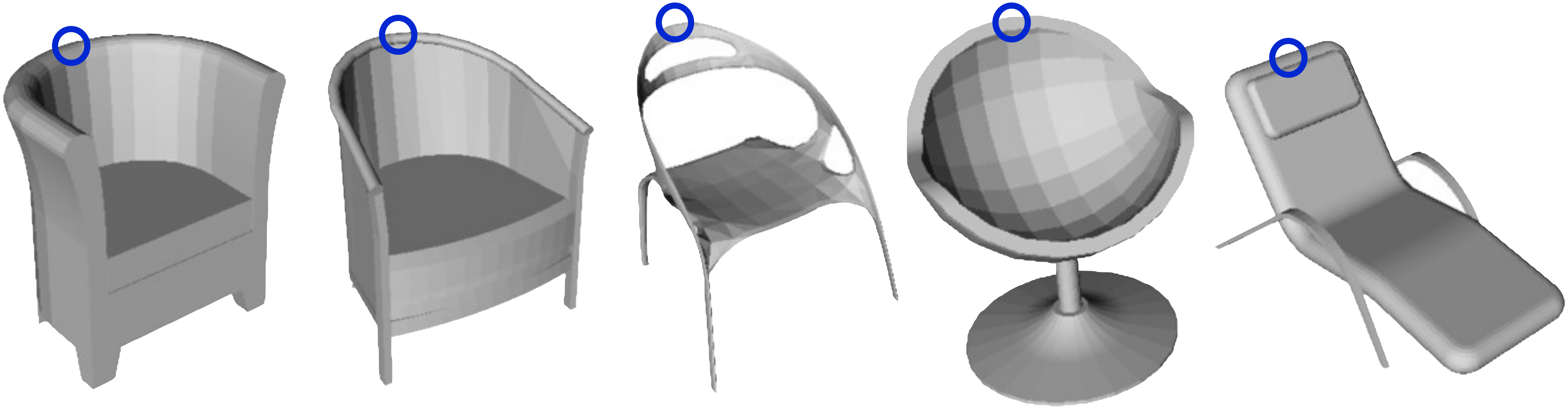
3. Deformable Template

Complexity: $O(N^{1.5})$

Goal

Find point correspondences for all pairs of models in collections with large geometric variations

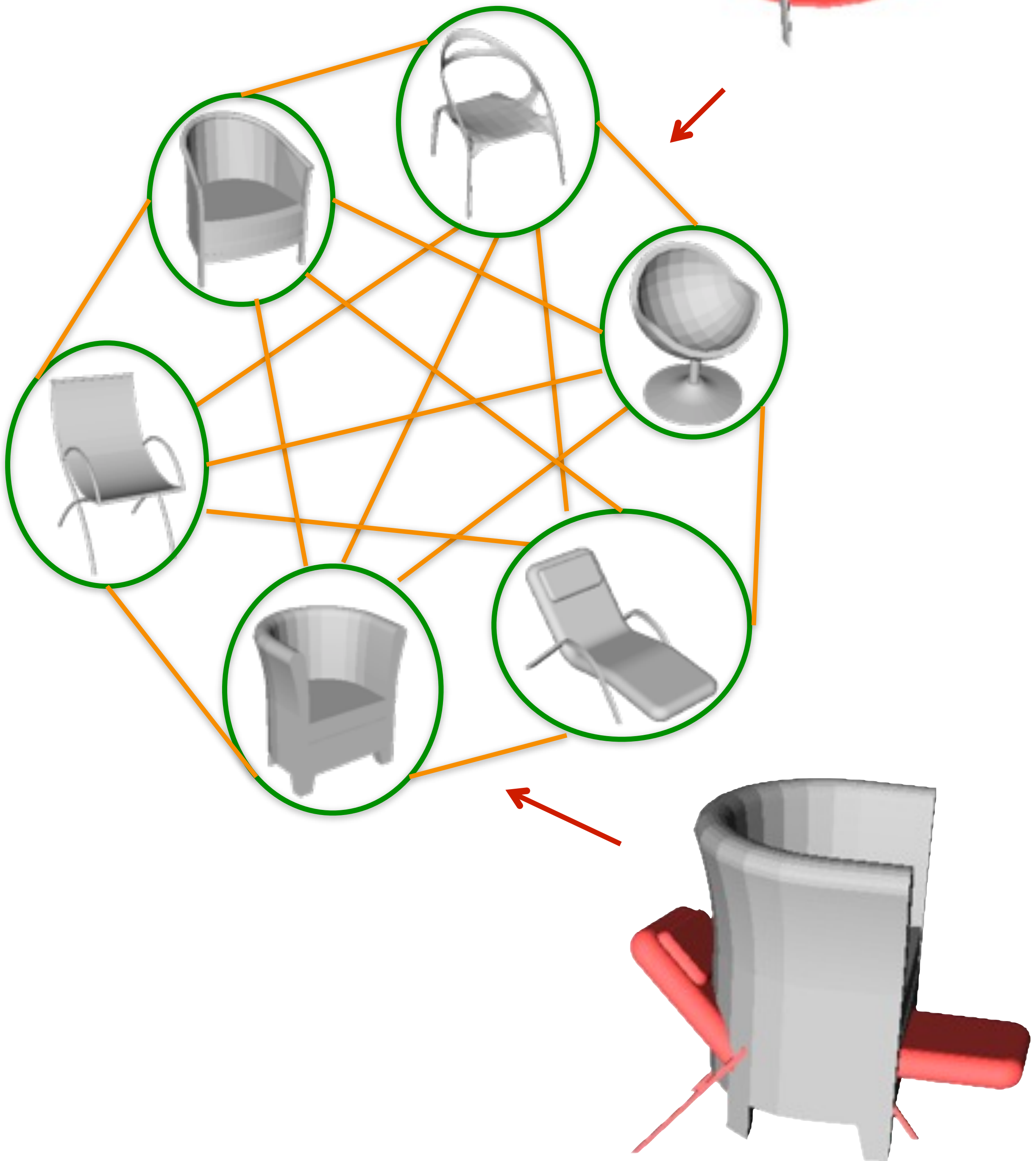
- Diverse shapes
- Efficient computation



Previous Work

Correspondences in a collection

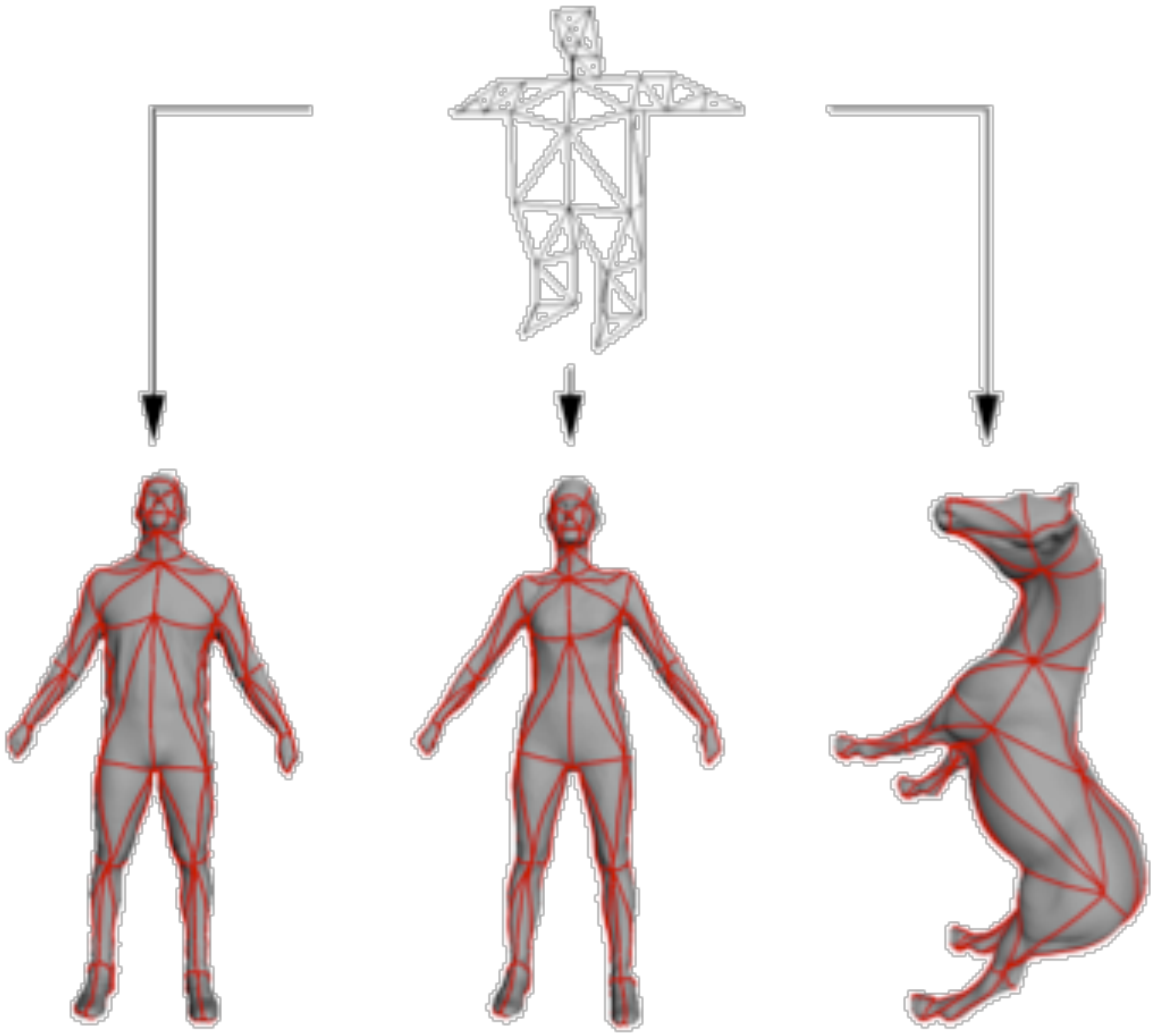
- ➡ All pairwise alignments
- o Template fitting
- o Map optimization



Previous Work

Correspondences in a collection

- All pairwise alignments
- ➡ Template fitting
- Map optimization

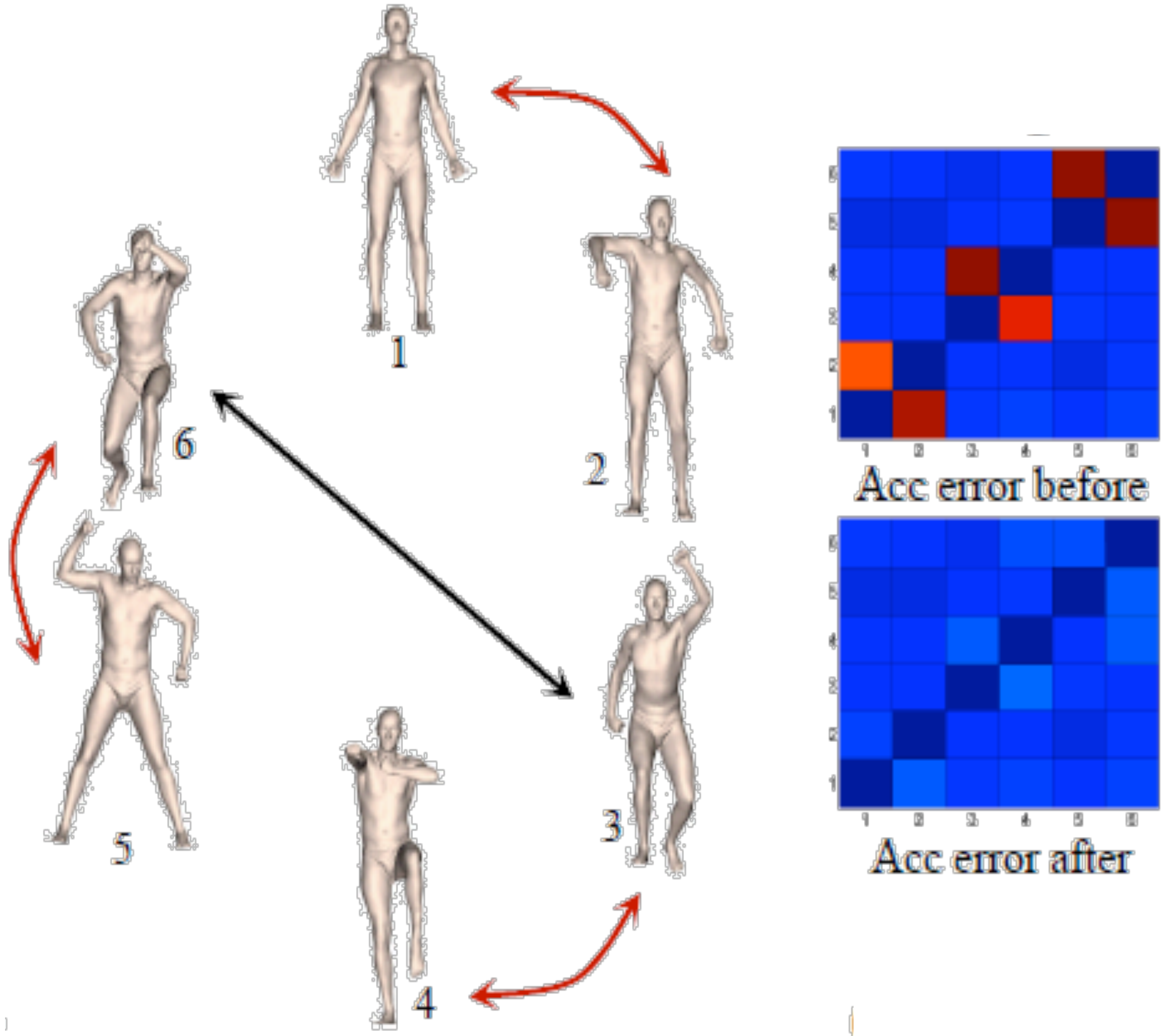


Praun et al., SIGGRAPH'01

Previous Work

Correspondences in a collection

- All pairwise alignments
- Template fitting
- ➡ Map optimization



Nguyen et al., SGP’11

Challenges

Efficient matching of diverse collections

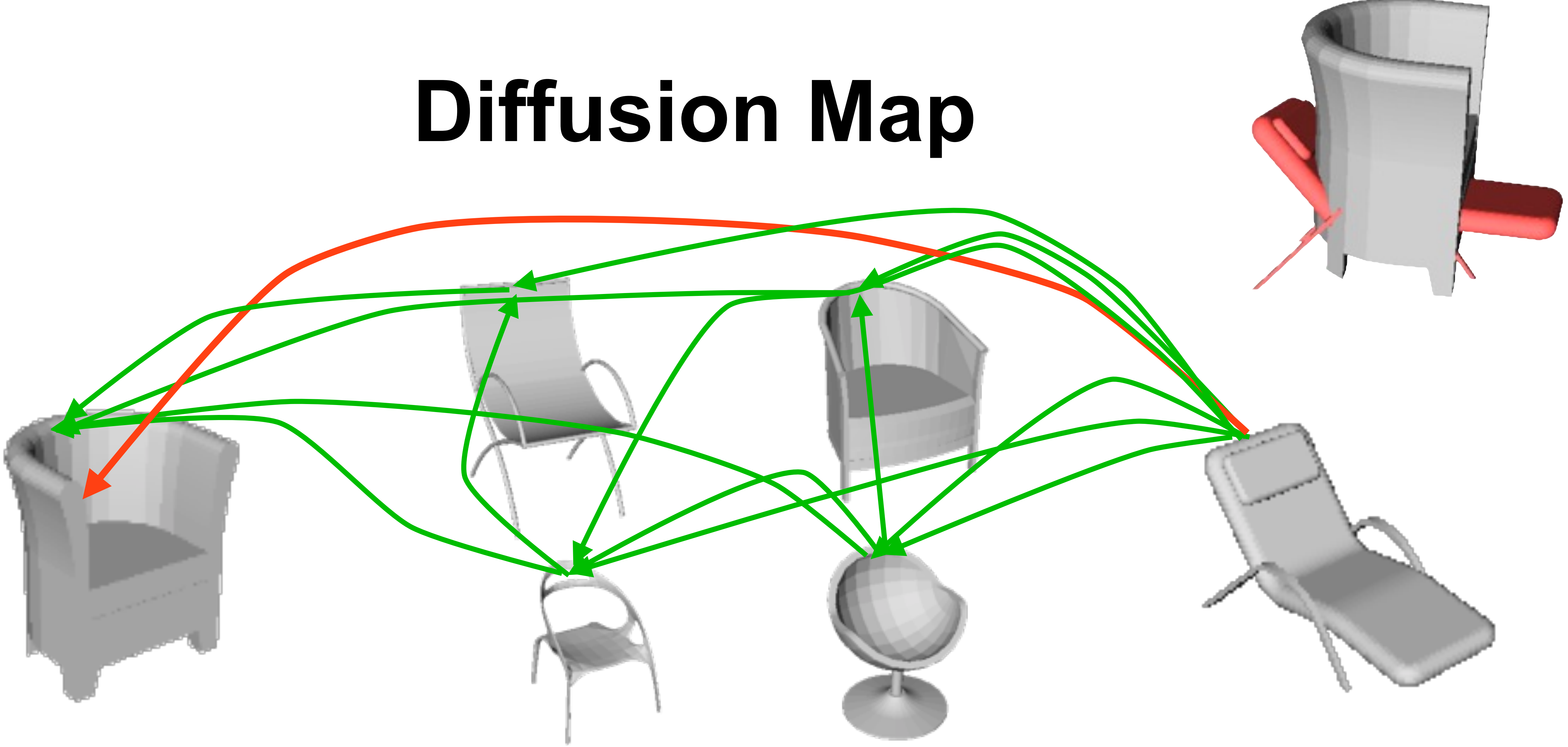
- Geometric alignment only works for similar shapes
- Point-to-point correspondences do not handle ambiguity
- Computing all pairwise alignments is $O(N^2)$

Our Approach

Efficient matching of diverse collections

- Geometric alignment only works for similar shapes
- Point-to-point correspondences do not handle ambiguity
- Computing all pairwise alignments is O(N^2)

Diffusion Map

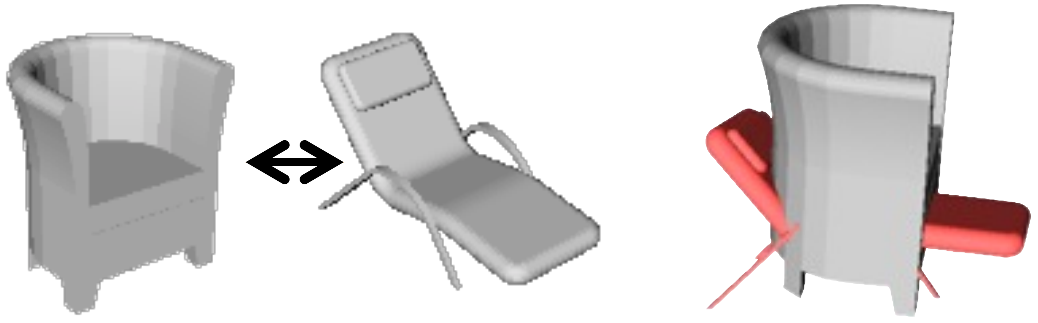


Our Approach

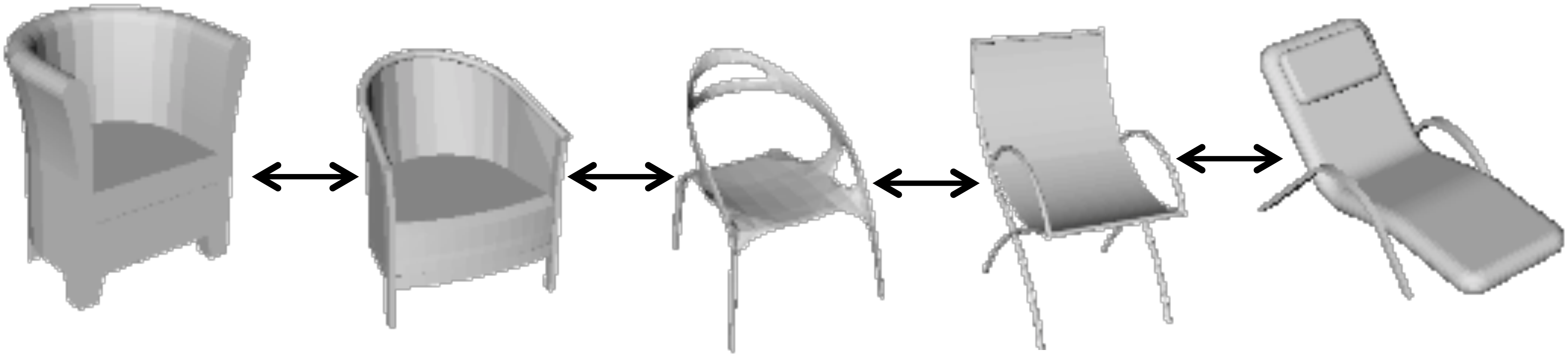
Efficient matching of diverse collections

- ➡ Geometric alignment only works for similar shapes
- ○ Point-to-point correspondences do not handle ambiguity
- ○ Computing all pairwise alignments is $O(N^2)$

Traditional correspondence:



Diffusion map leverages transitivity:

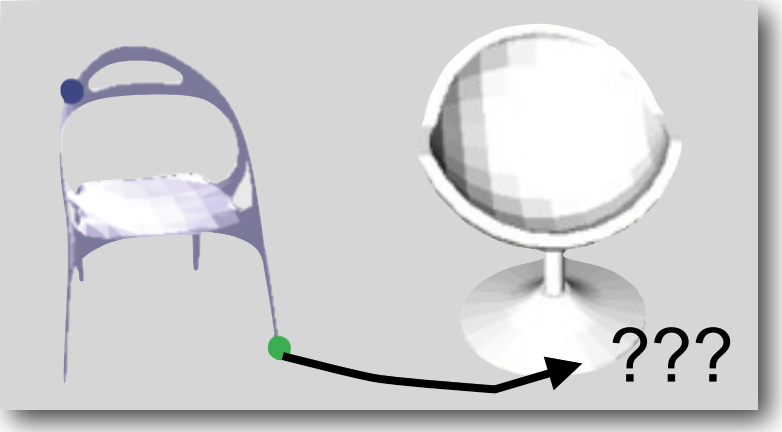


Our Approach

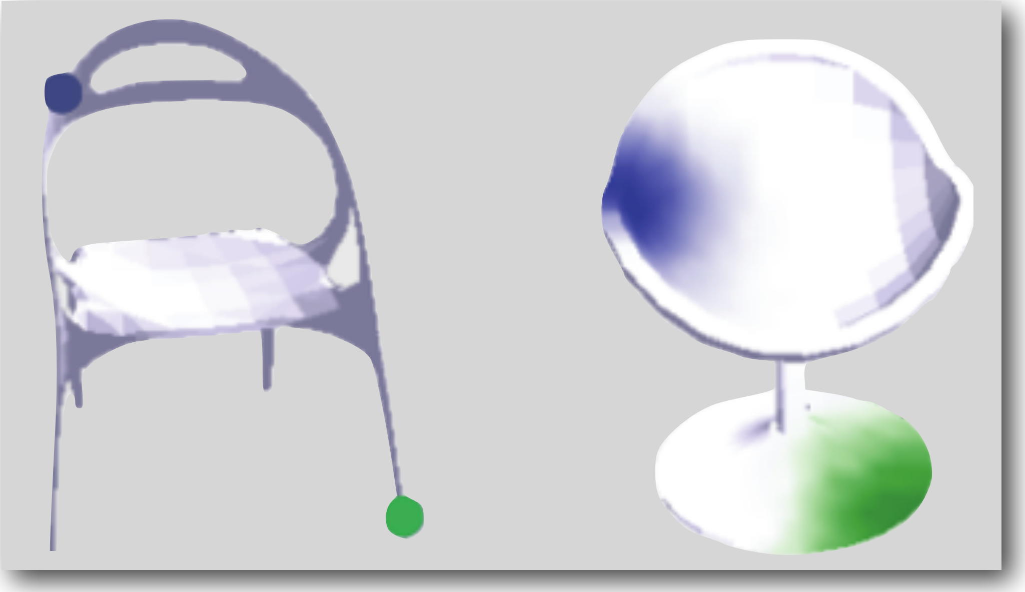
Efficient matching of diverse collections

- Geometric alignment only works for similar shapes
- ➡ Point-to-point correspondences do not handle ambiguity
- Computing all pairwise alignments is O(N^2)

Traditional correspondence:



Diffusion map produces continuous similarity values:



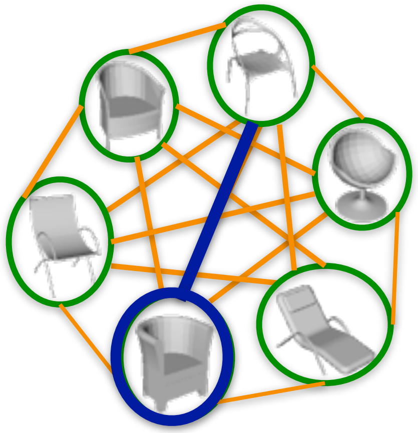
Our Approach

Efficient matching of diverse collections

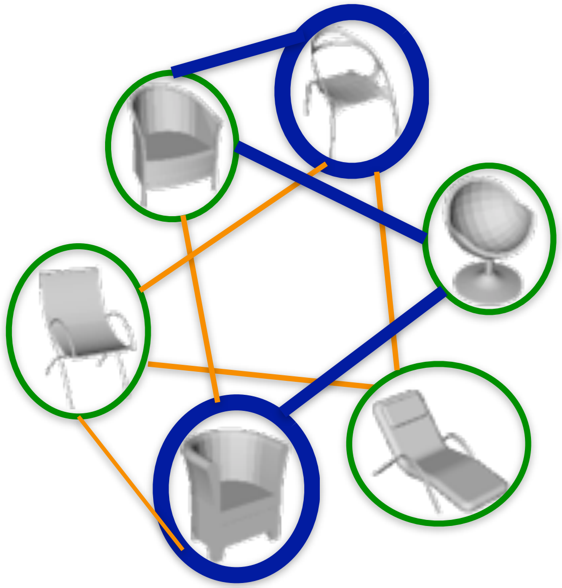
- Geometric alignment only works for similar shapes
- Point-to-point correspondences do not handle ambiguity

➡ Computing all pairwise alignments is $O(N^2)$

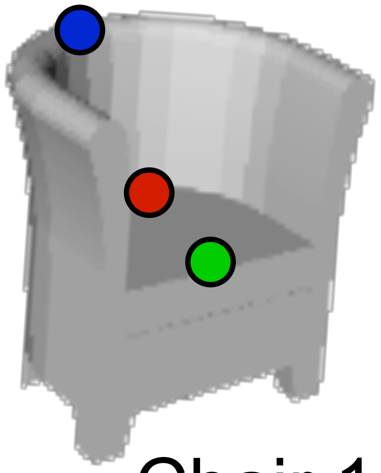
Traditional correspondence:



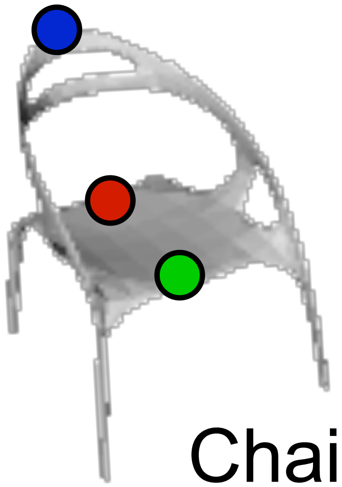
Diffusion map works with sparse alignments



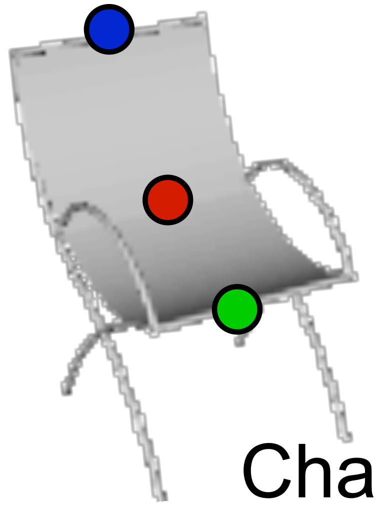
Diffusion Map



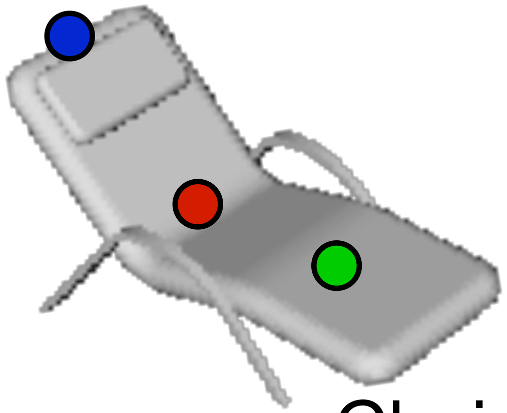
Chair 1



Chair 2

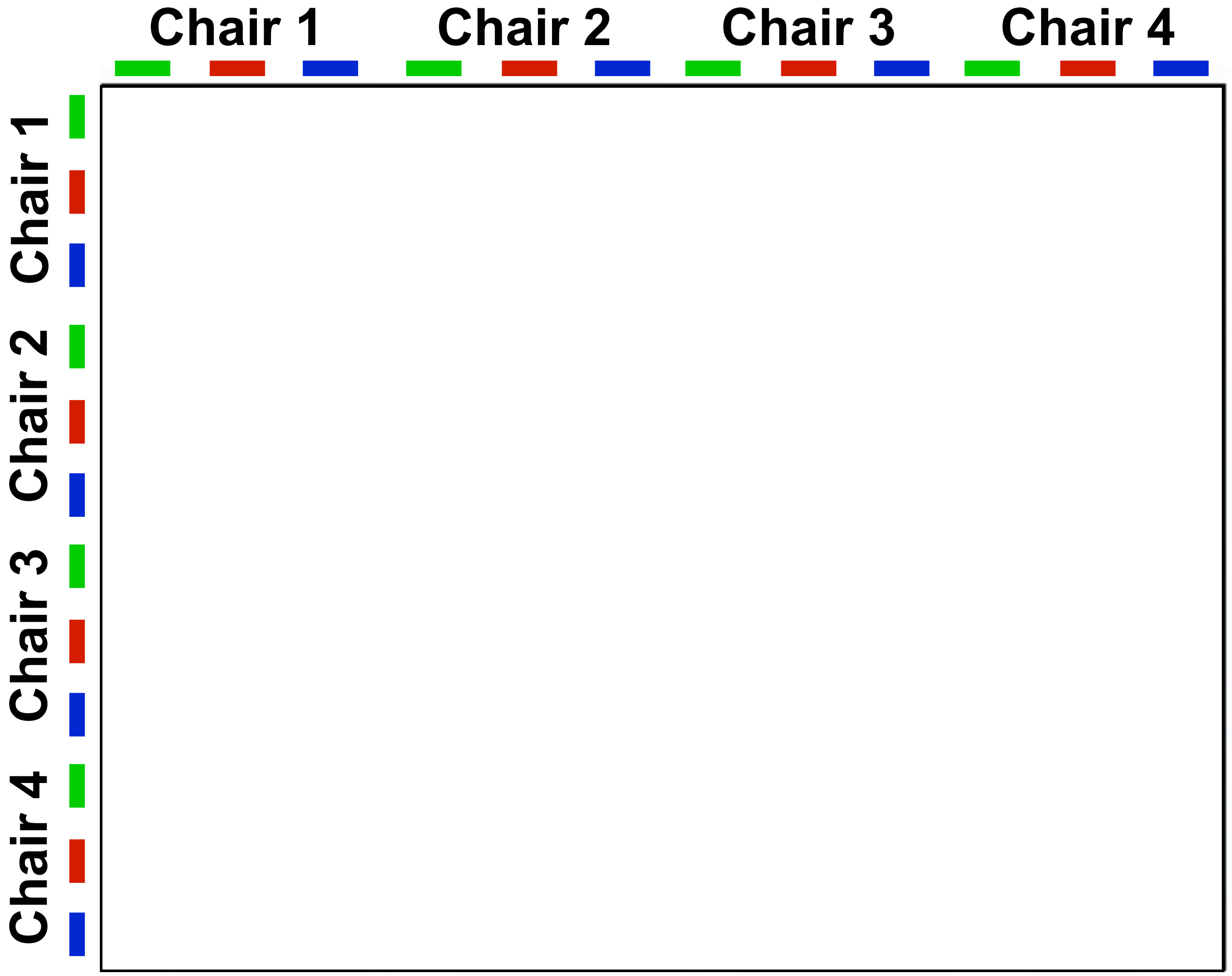


Chair 3

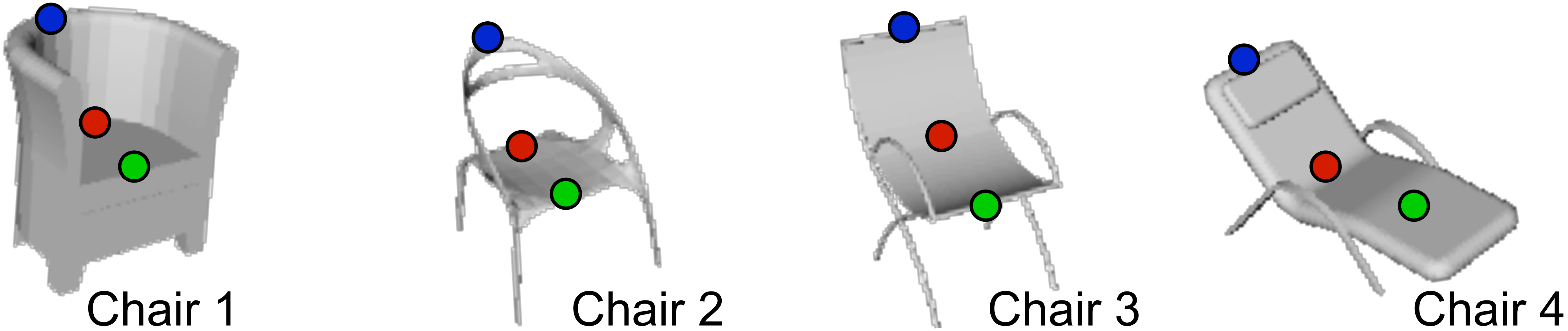


Chair 4

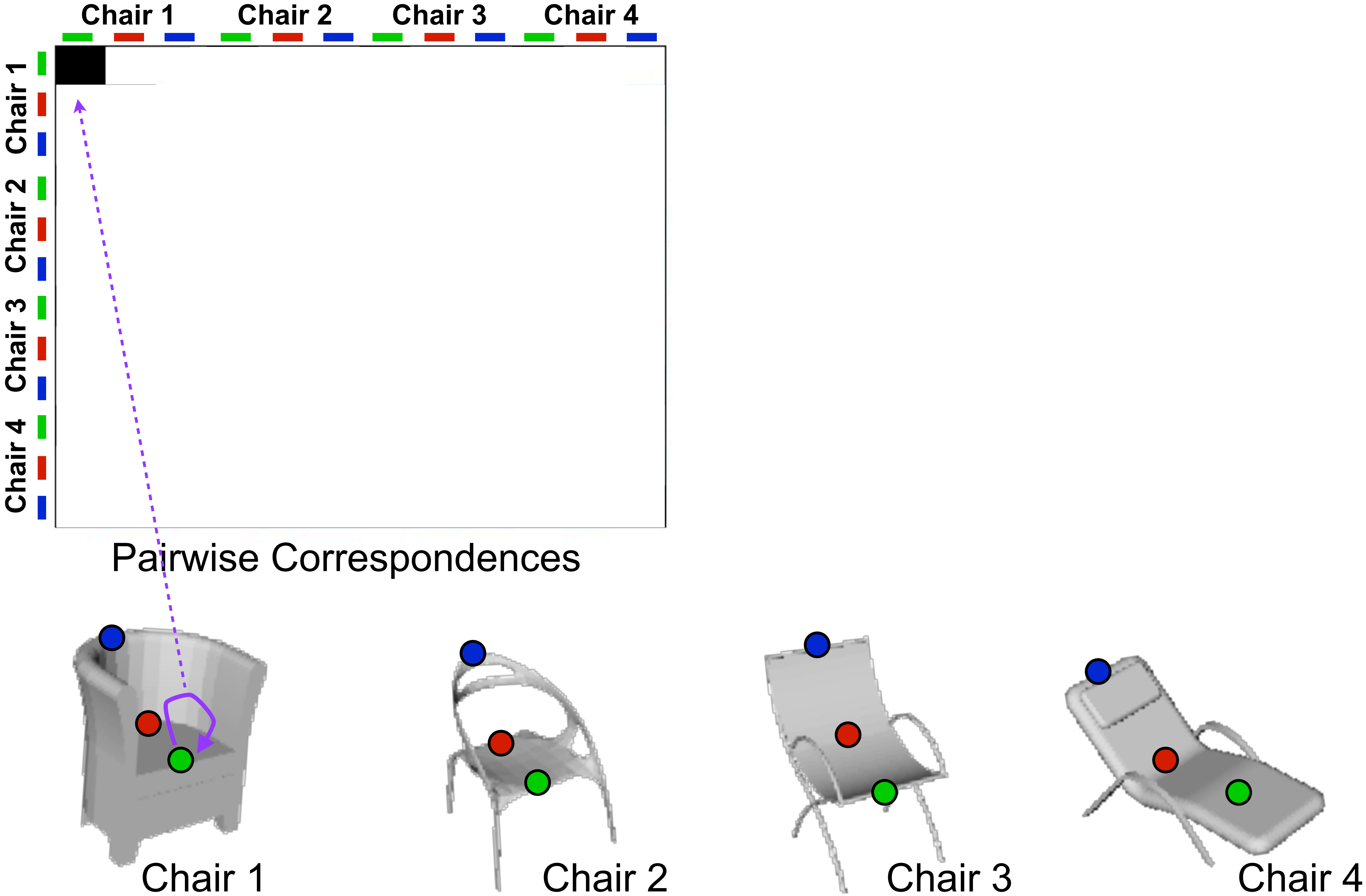
Diffusion Map



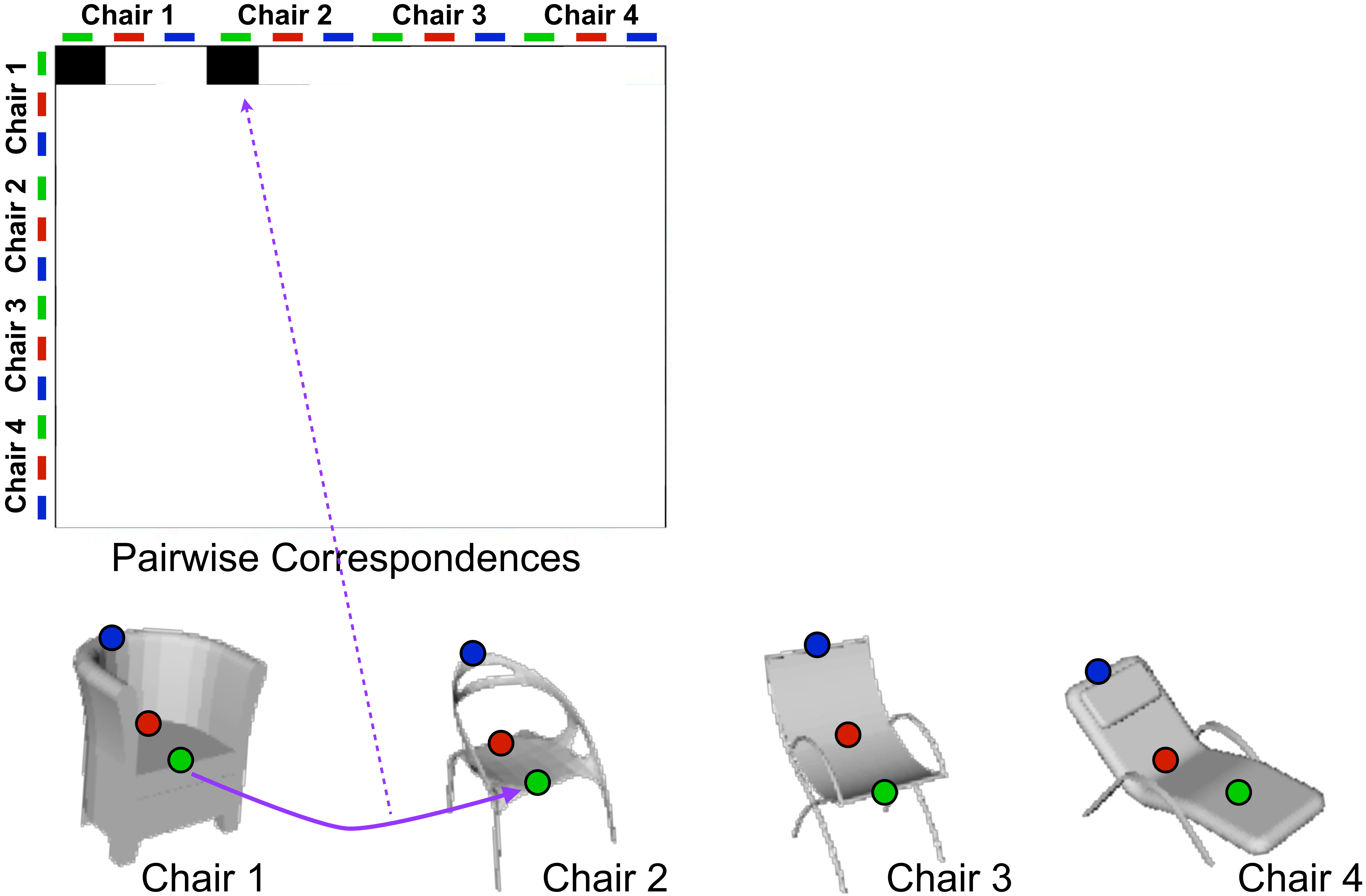
Pairwise Correspondences



Diffusion Map

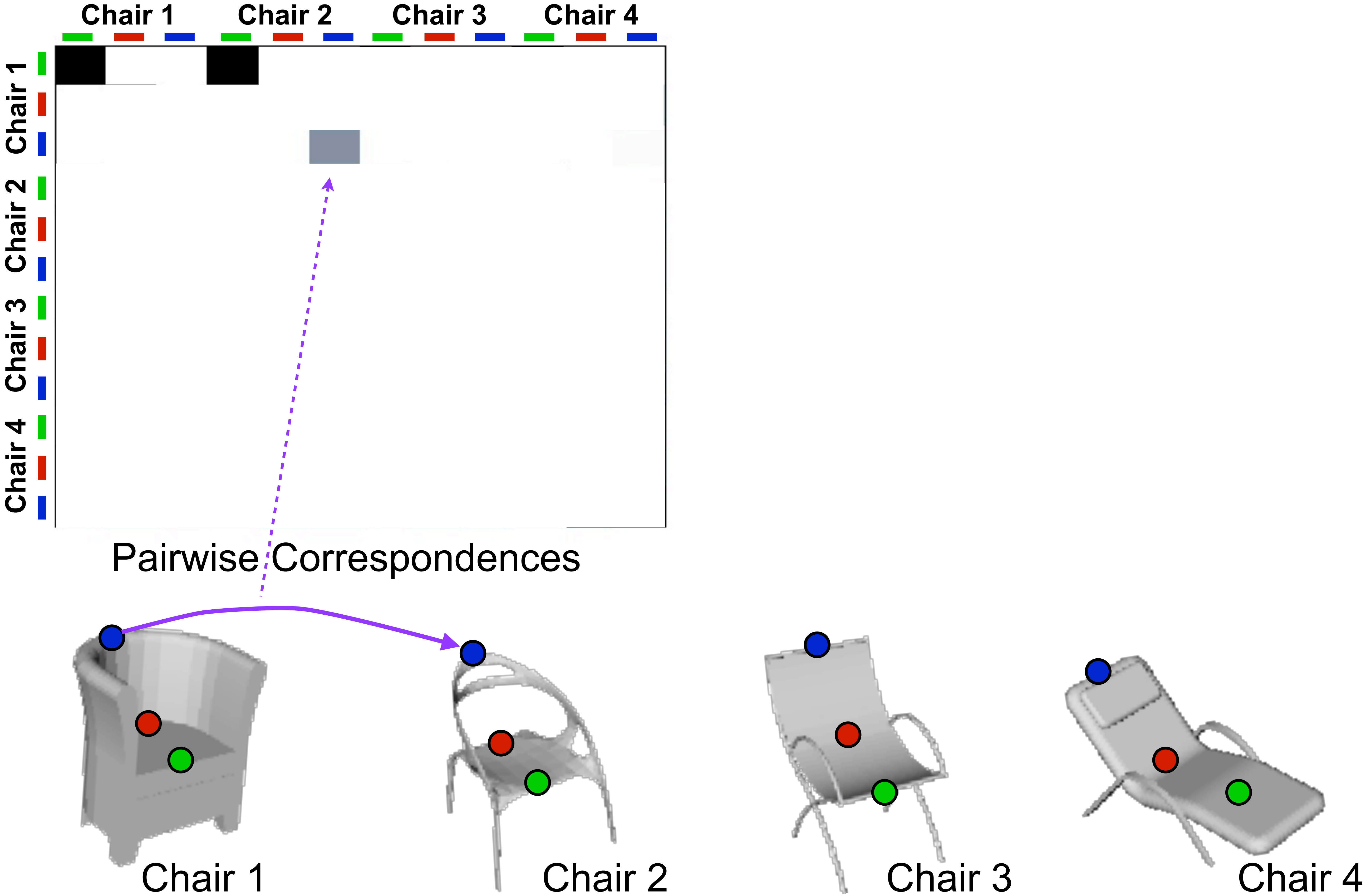


Diffusion Map



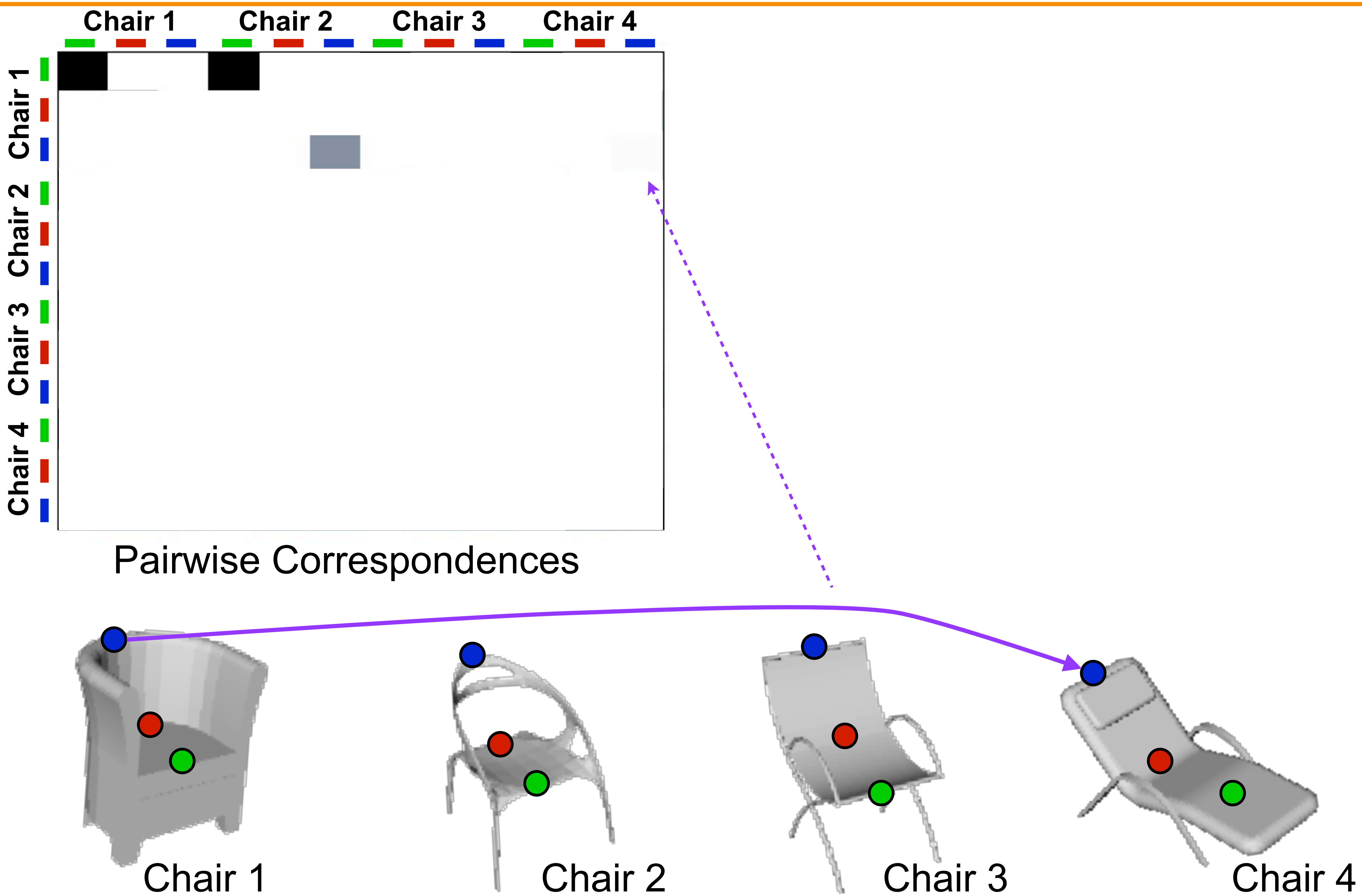
Pairwise Correspondences

Diffusion Map



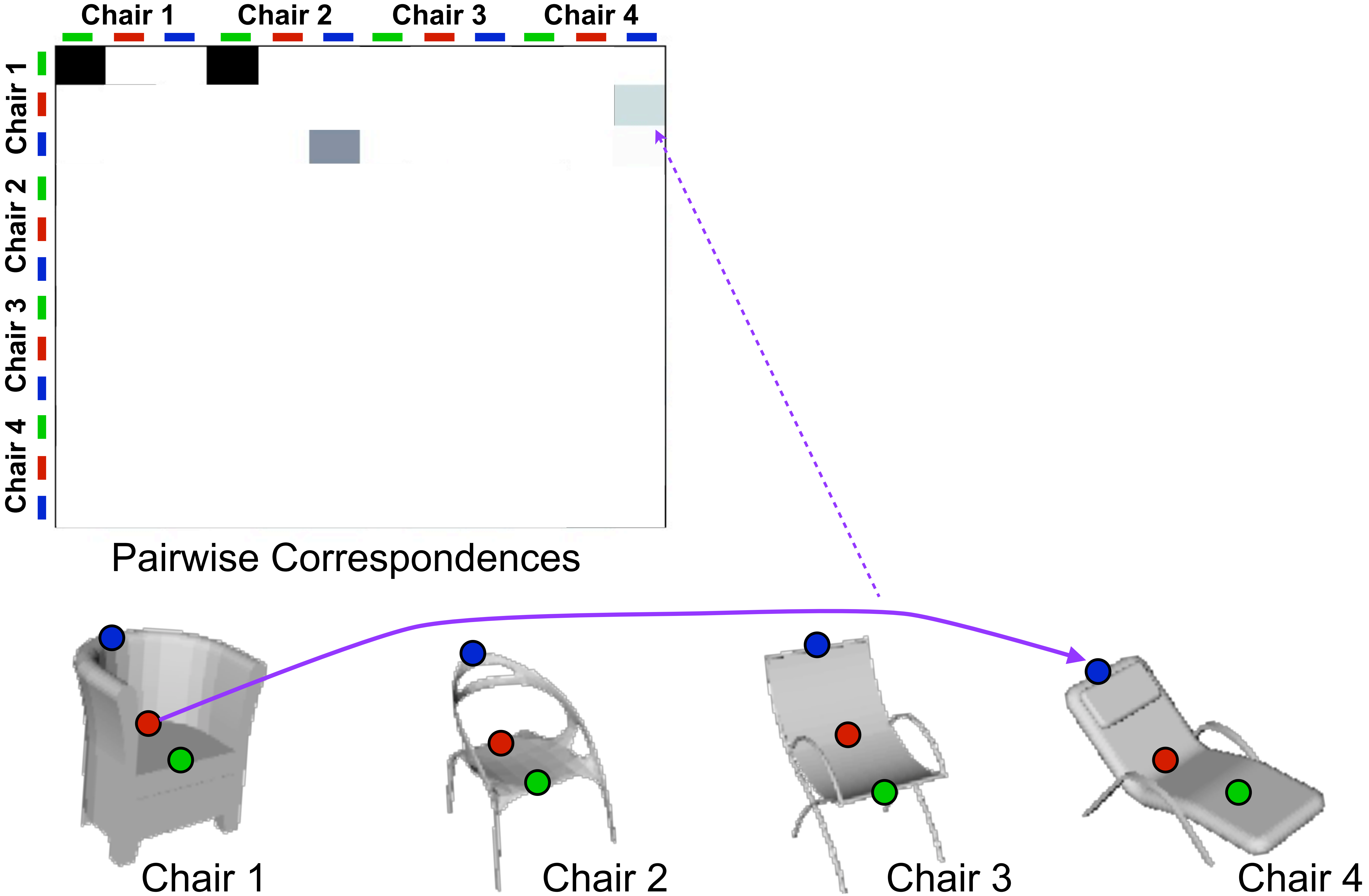
Pairwise Correspondences

Diffusion Map

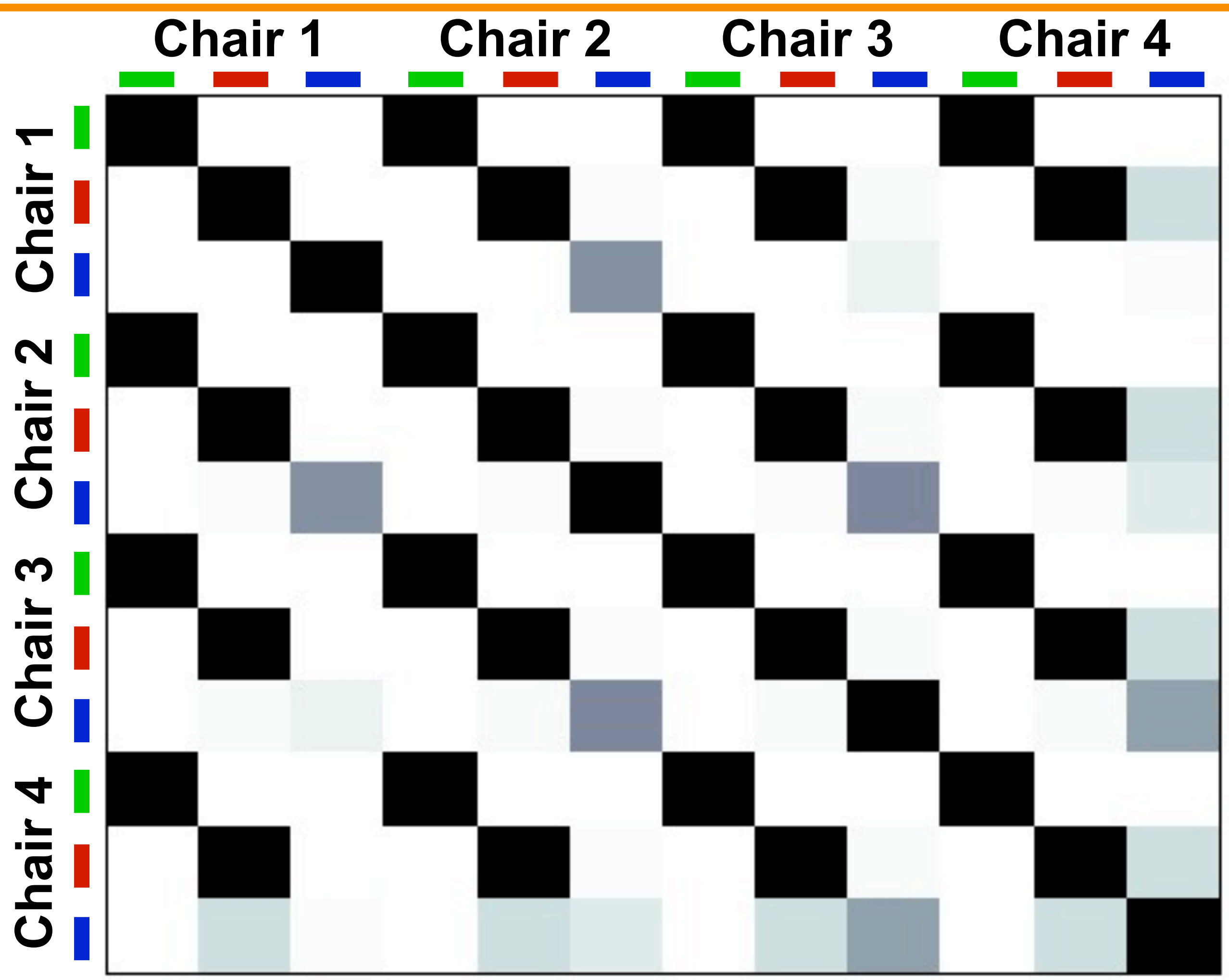


Pairwise Correspondences

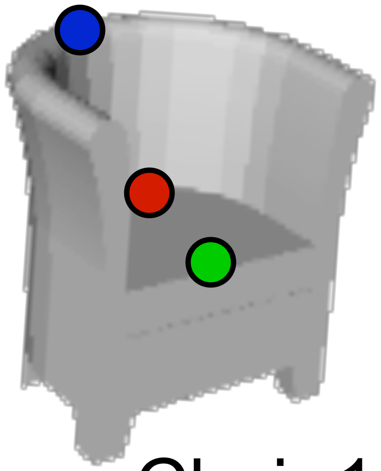
Diffusion Map



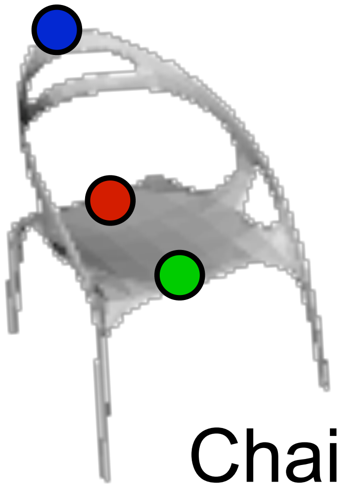
Diffusion Map



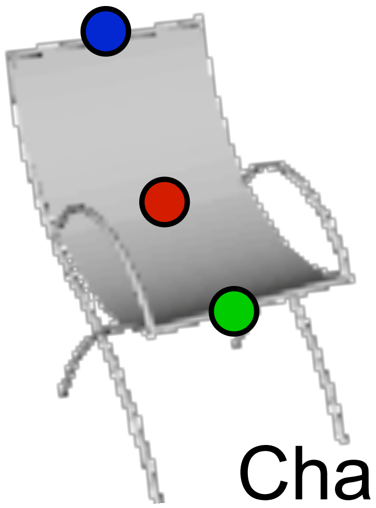
Pairwise Correspondences



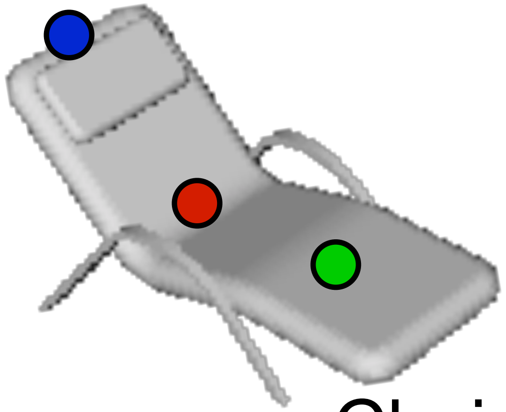
Chair 1



Chair 2

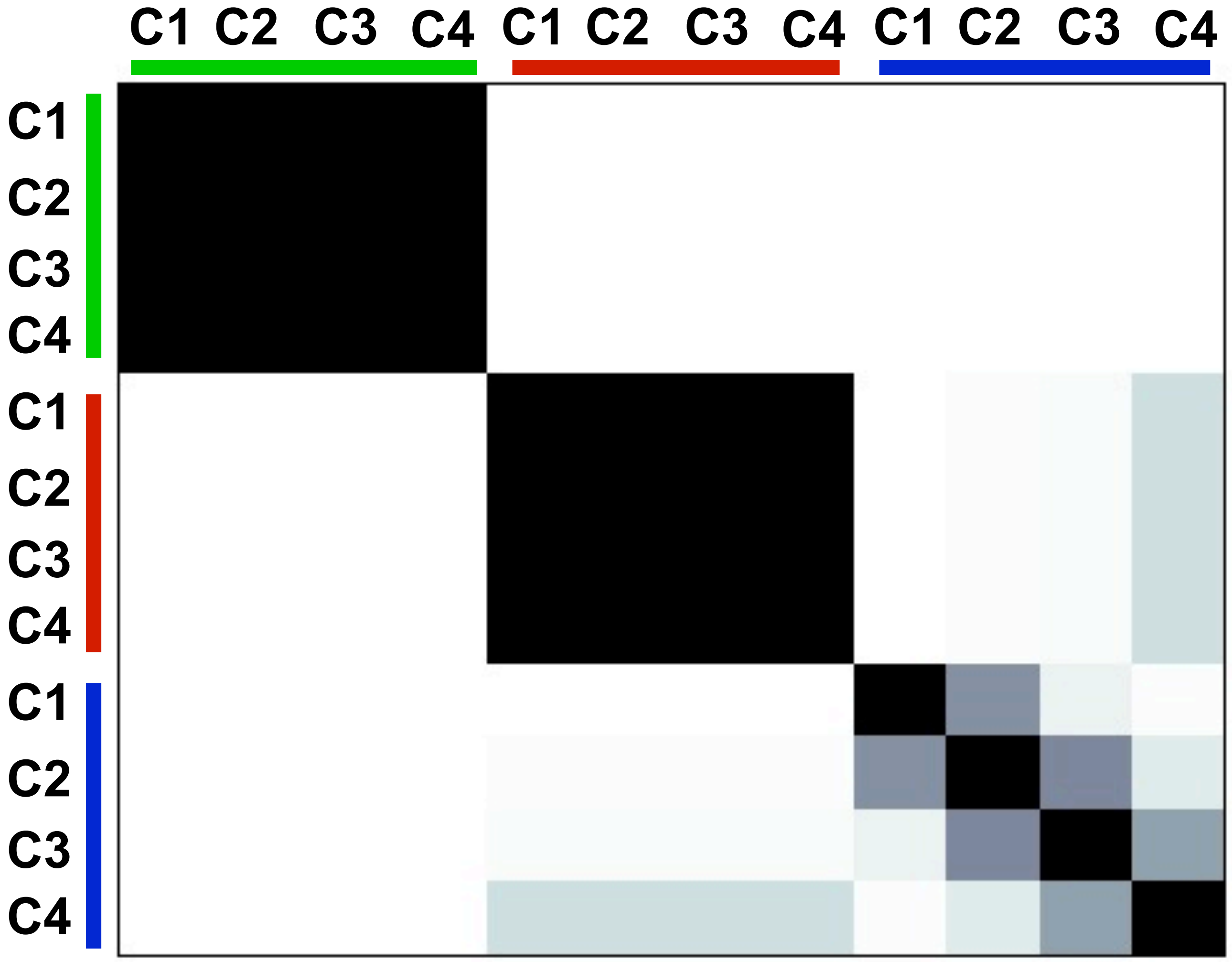


Chair 3

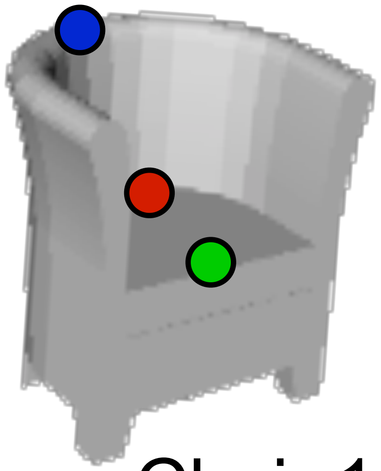


Chair 4

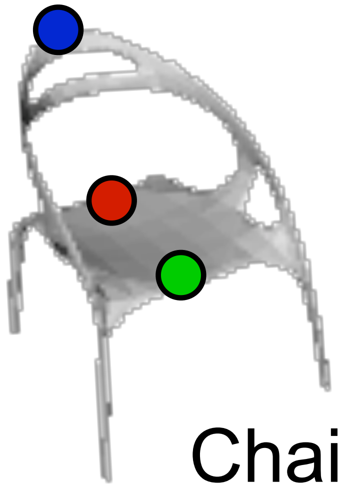
Diffusion Map



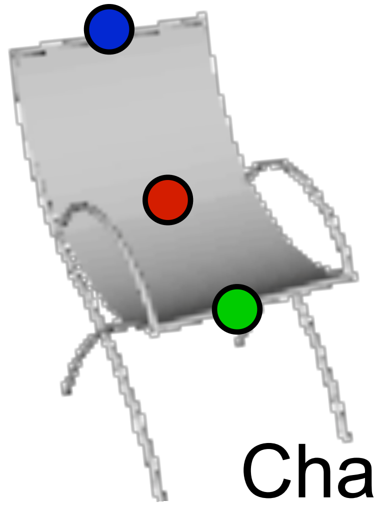
Pairwise Correspondences



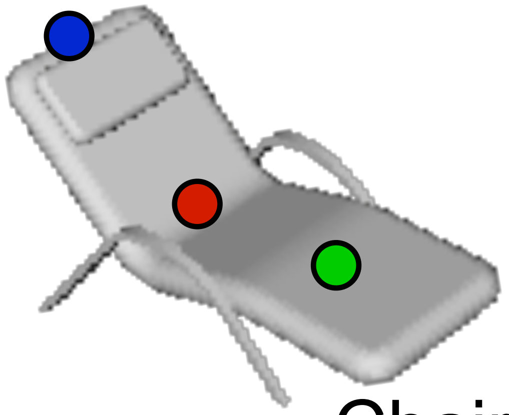
Chair 1



Chair 2

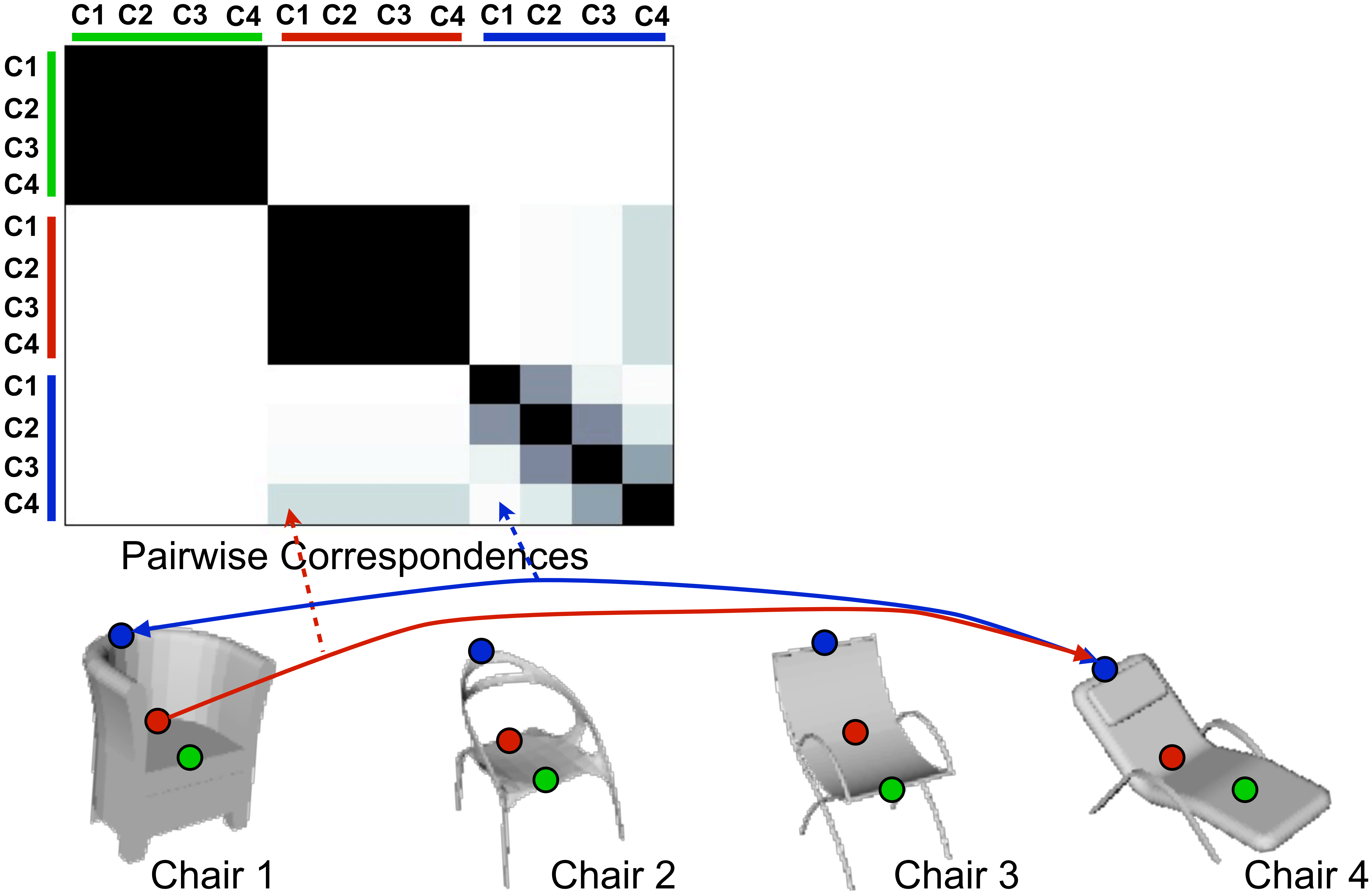


Chair 3

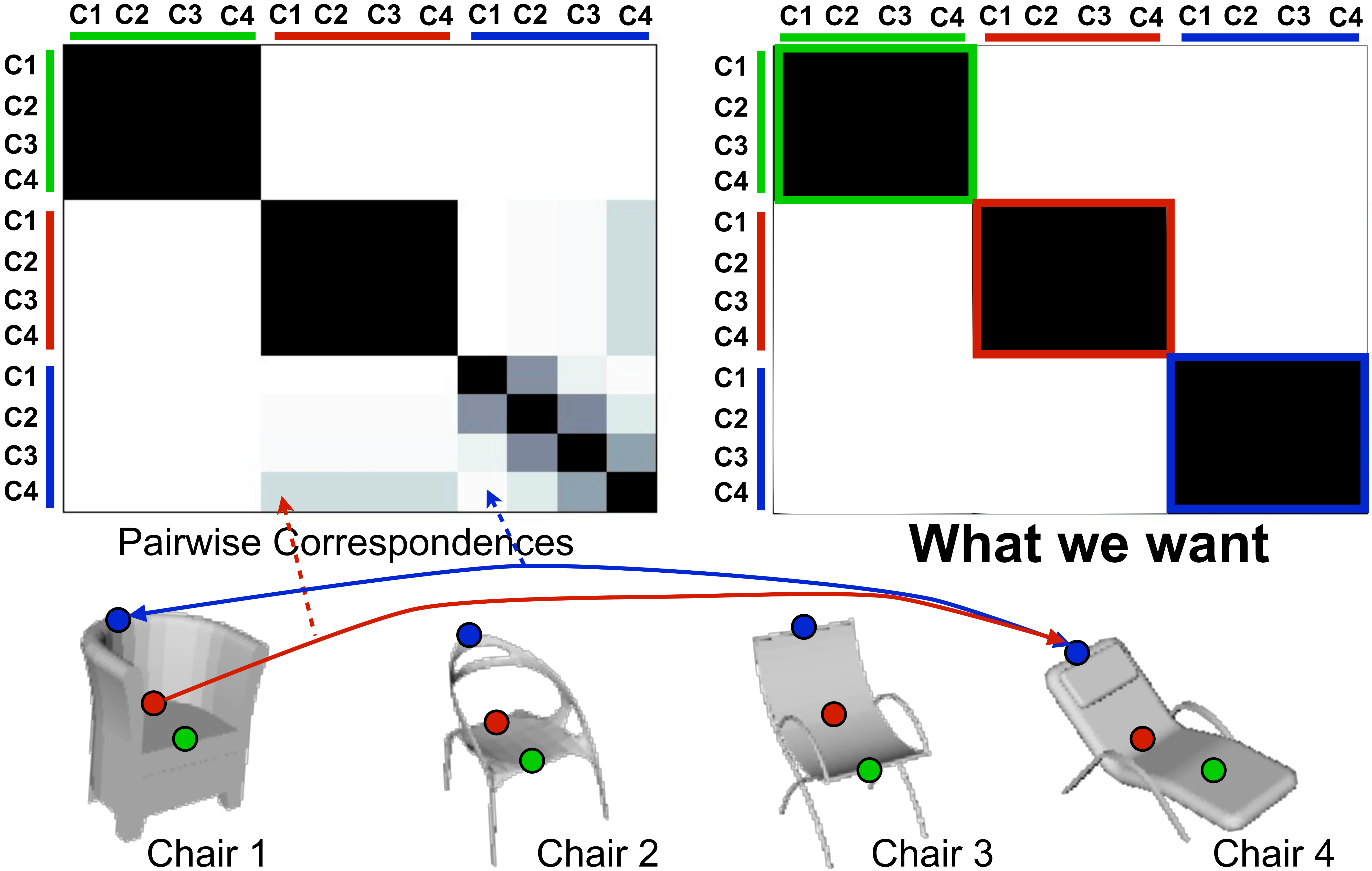


Chair 4

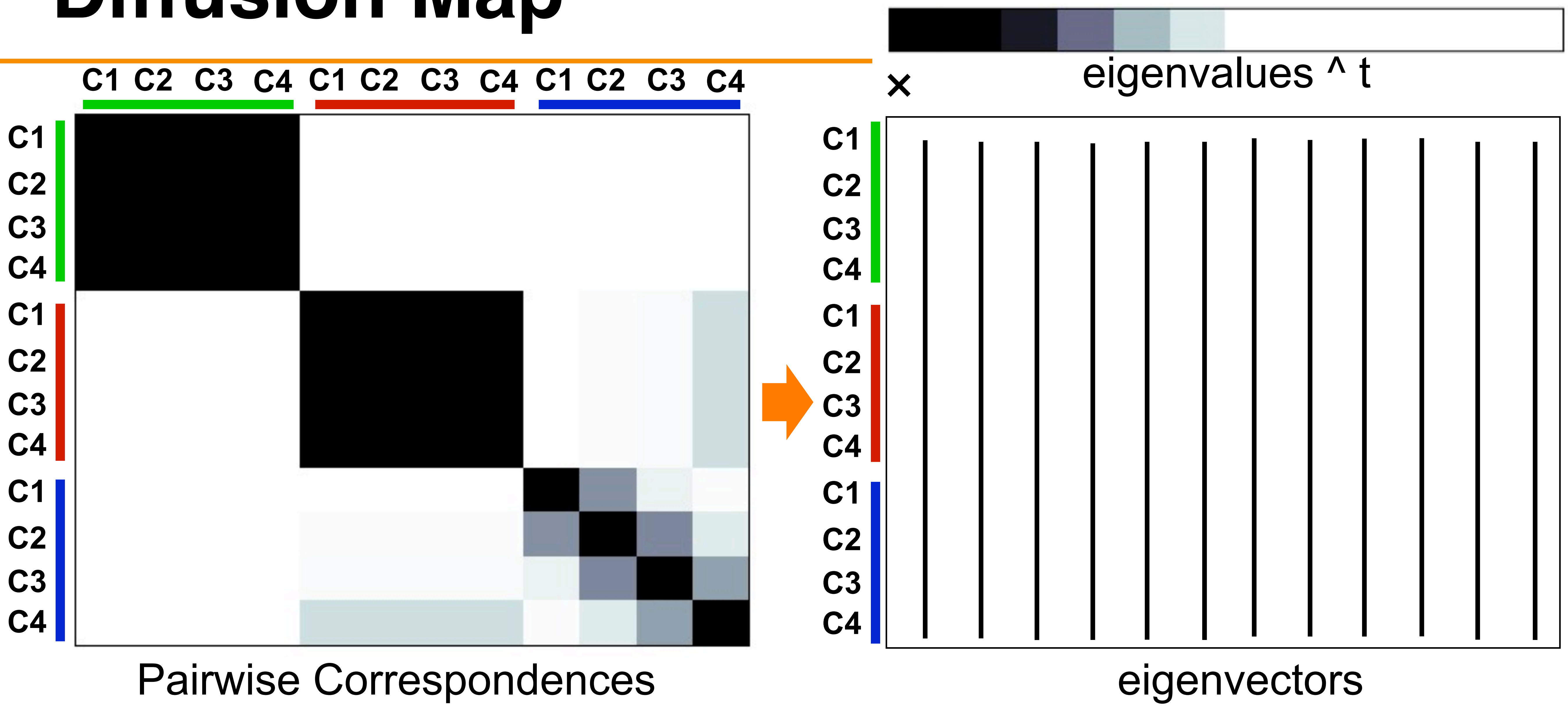
Diffusion Map

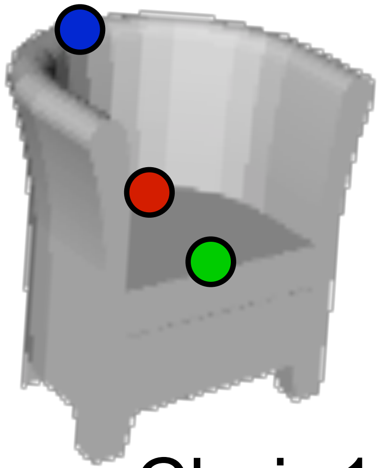


Diffusion Map

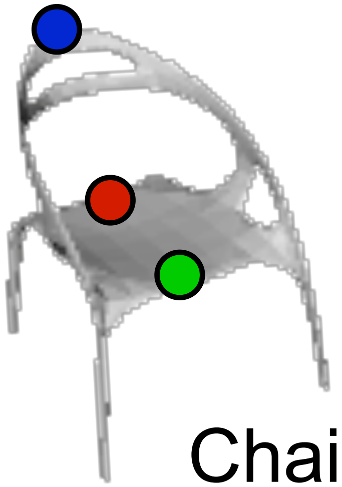


Diffusion Map

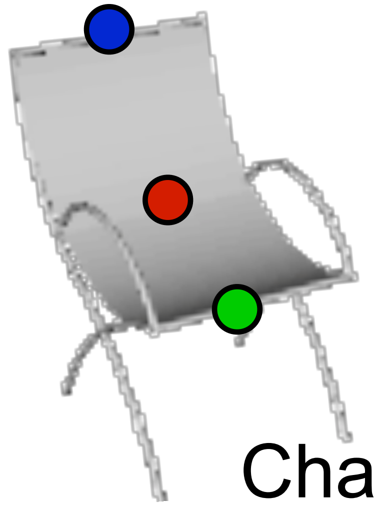




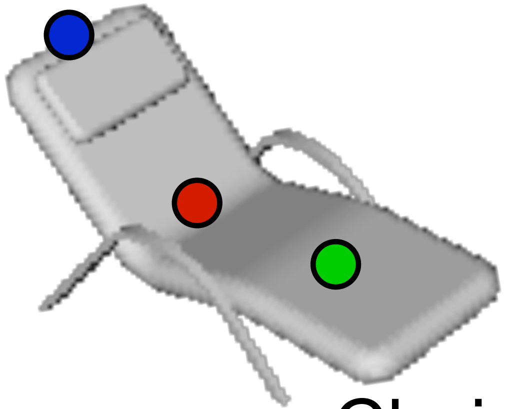
Chair 1



Chair 2

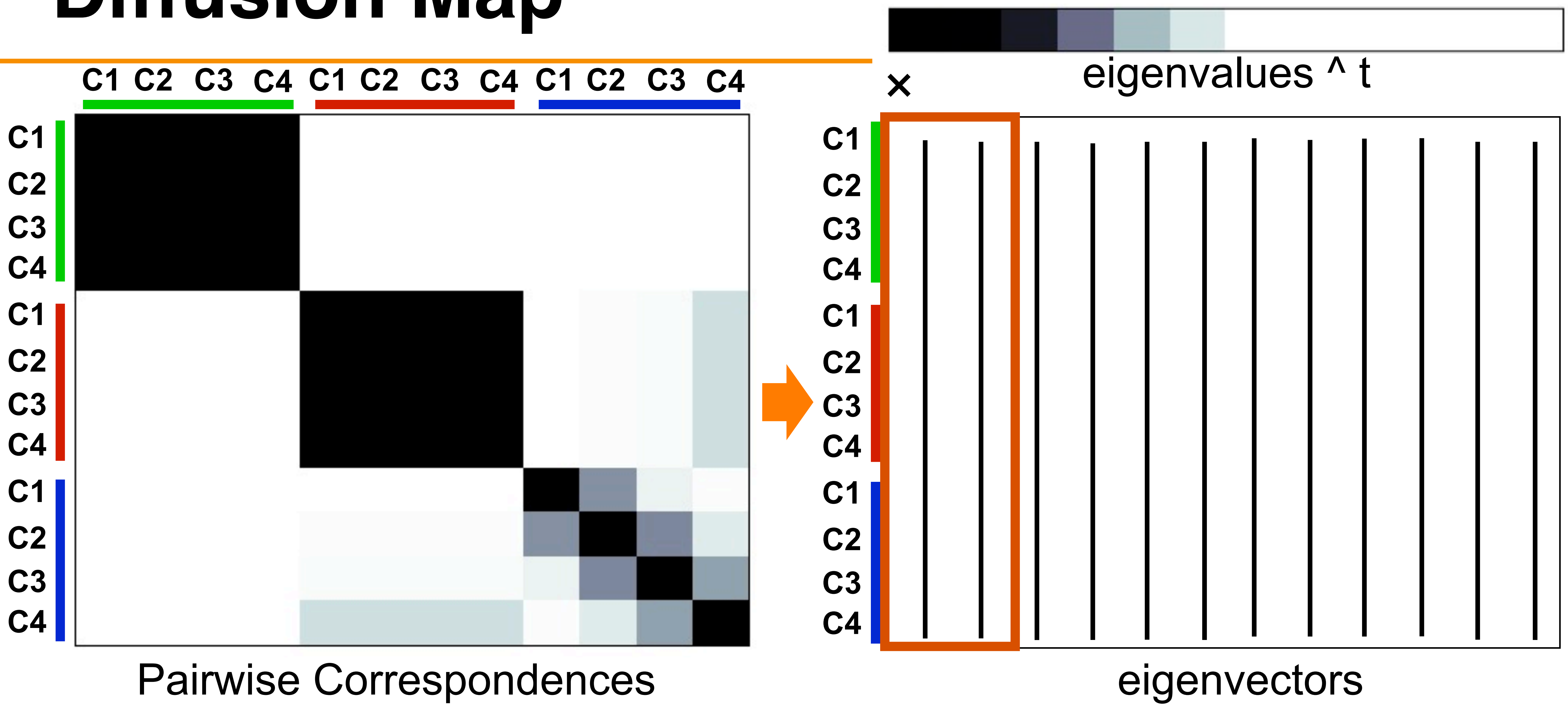


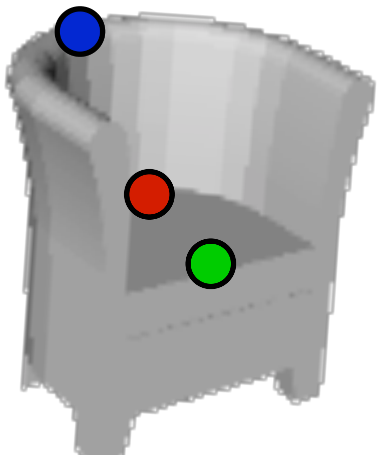
Chair 3

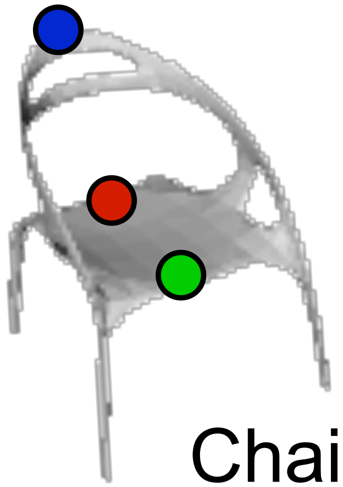


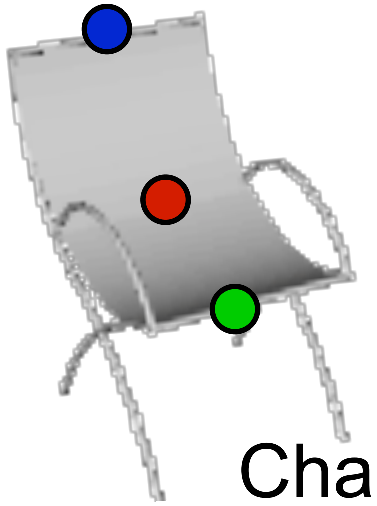
Chair 4

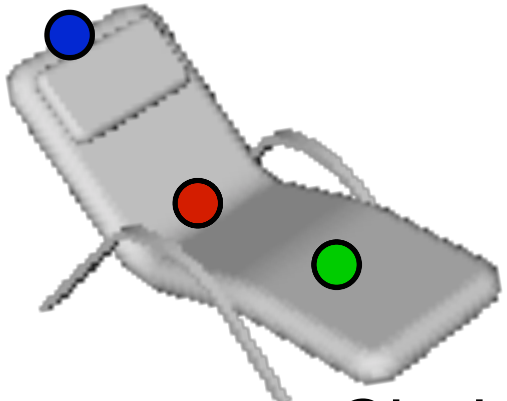
Diffusion Map



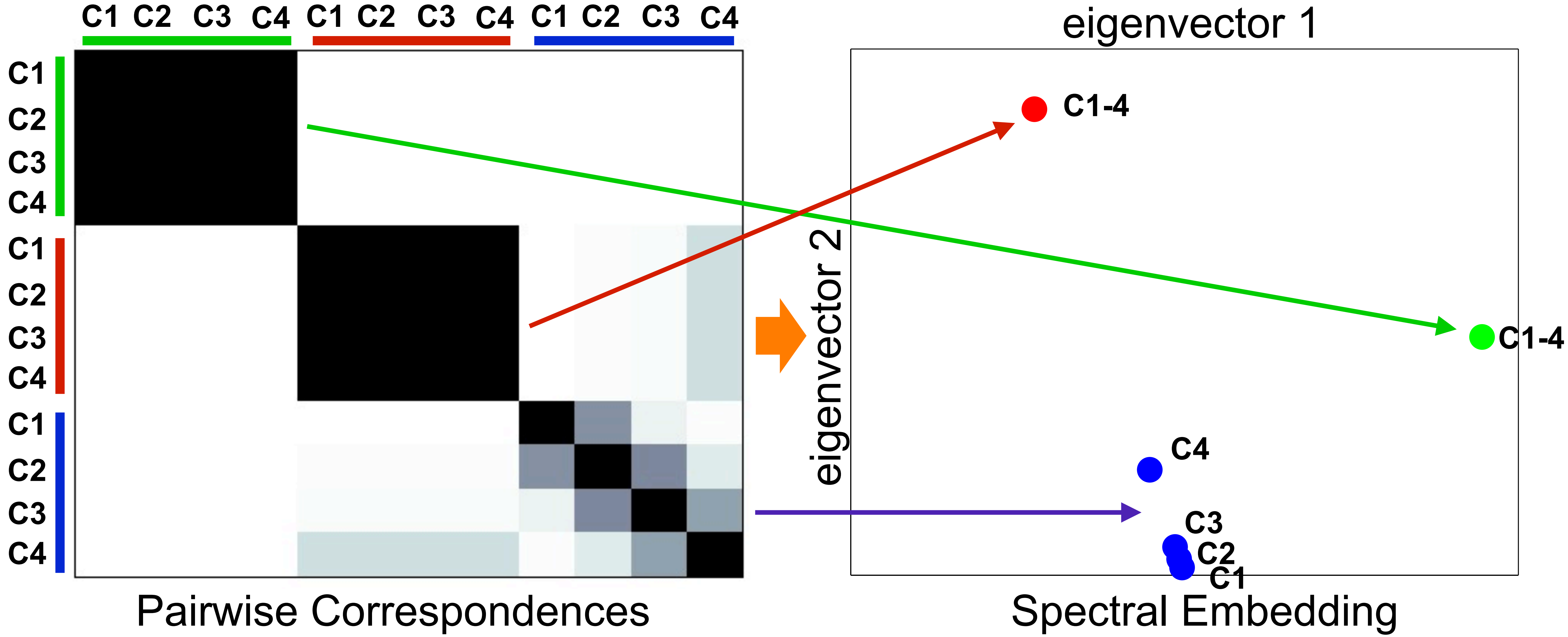

Chair 1

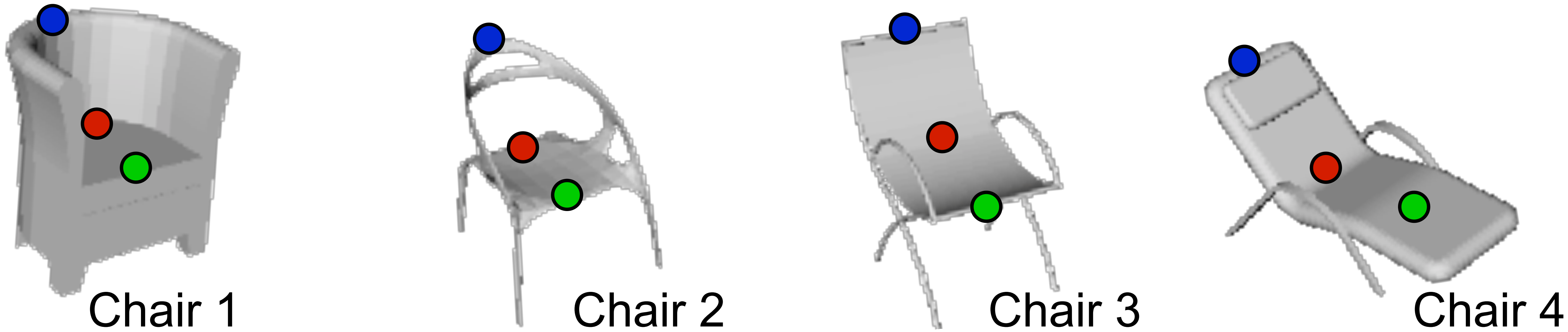

Chair 2


Chair 3

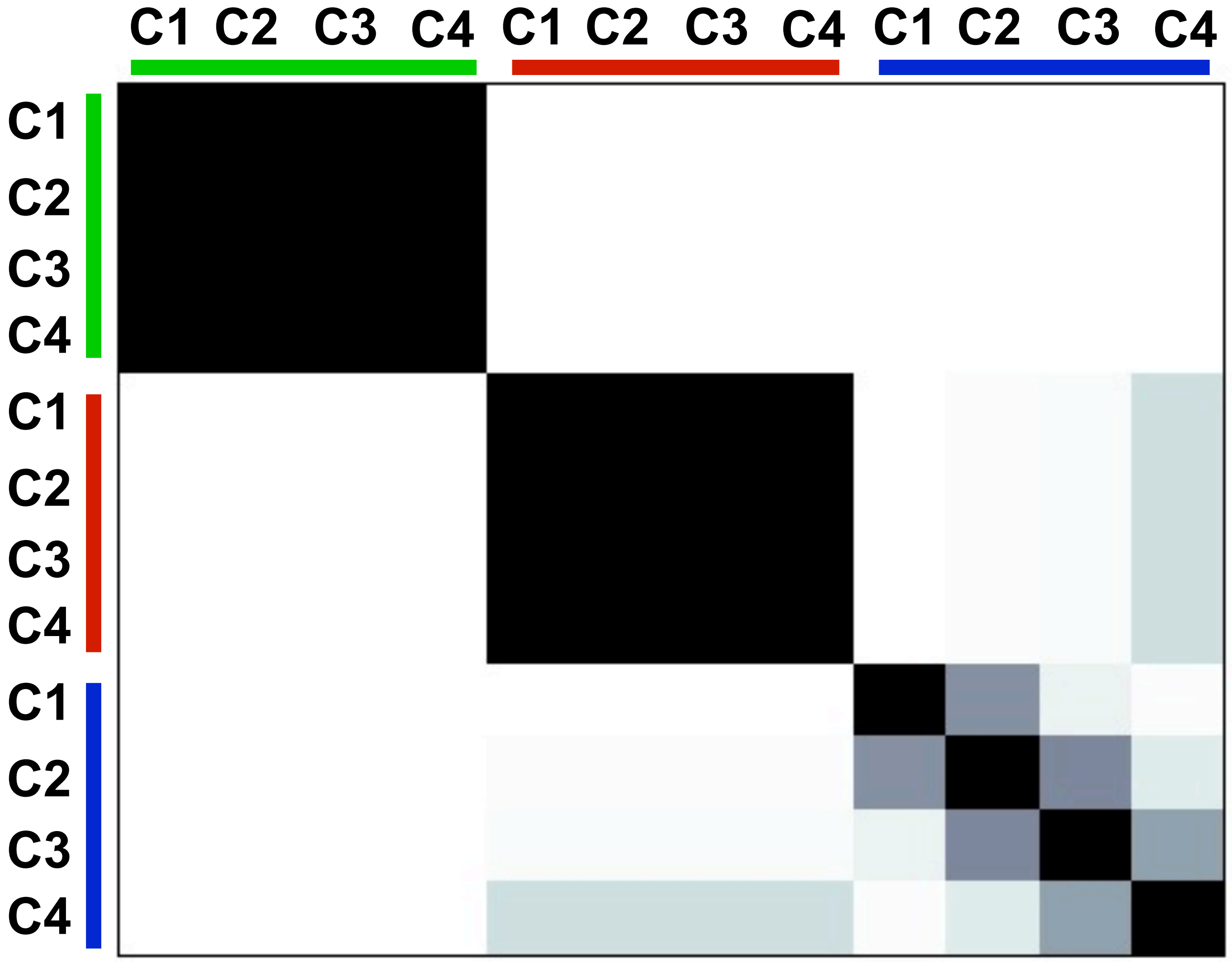

Chair 4

Diffusion Map

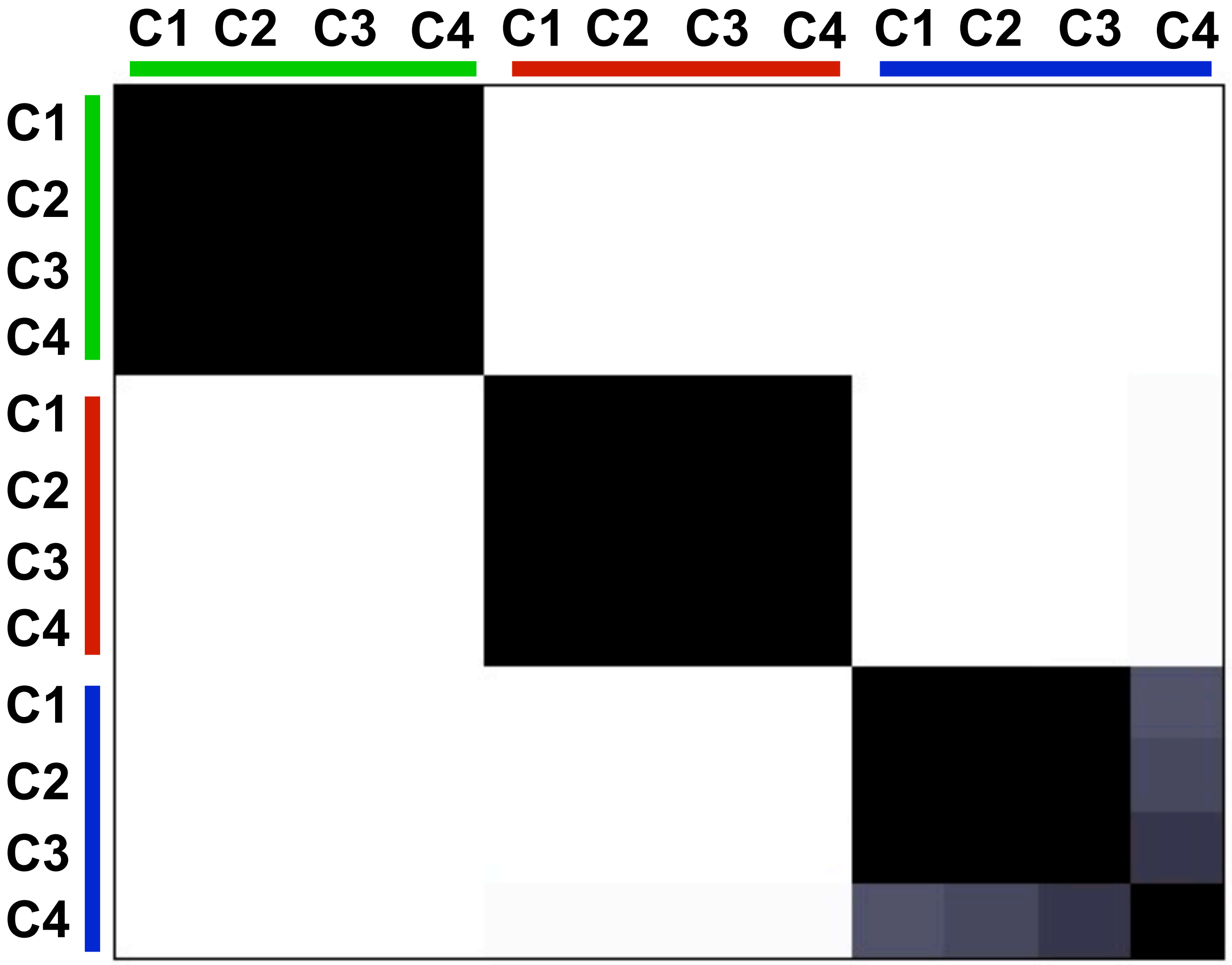




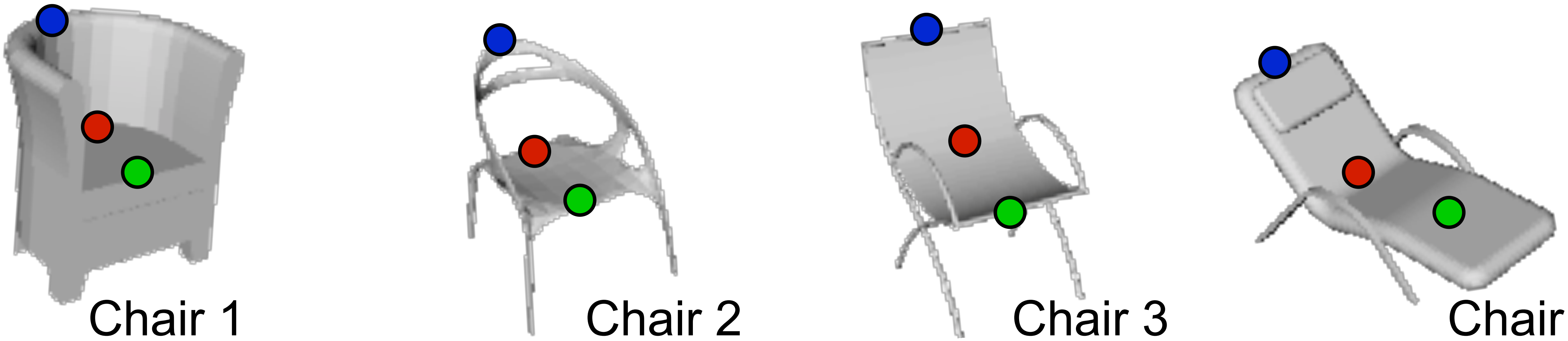
Diffusion Map



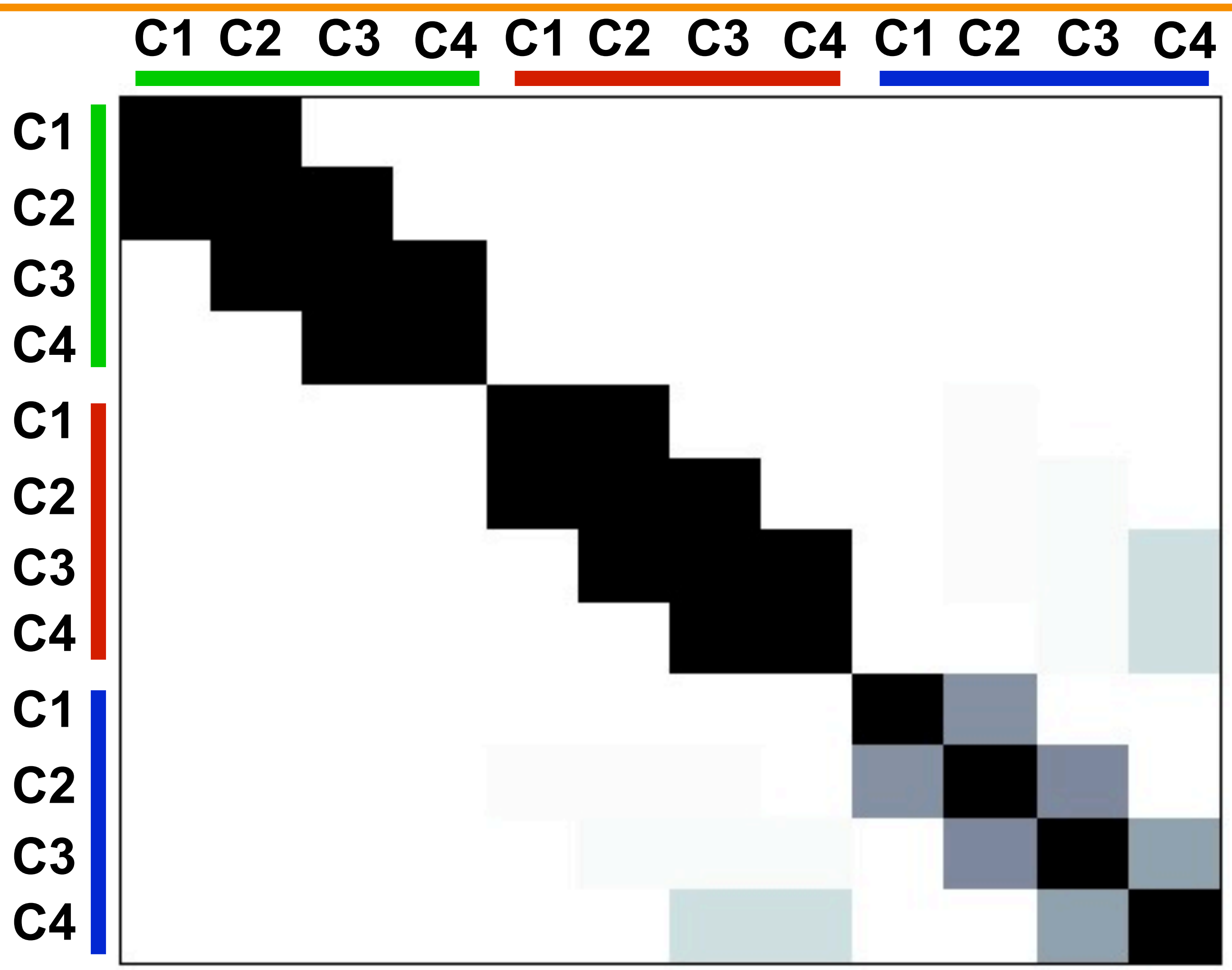
Pairwise Correspondences



Fuzzy Correspondences: $e^{\frac{-D_t(p_i,p_j)^2}{\sigma(p_i)^2}}$

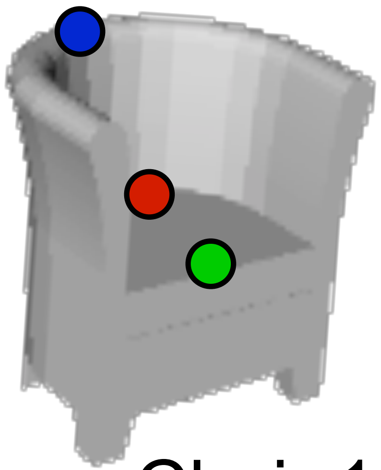


Diffusion Map

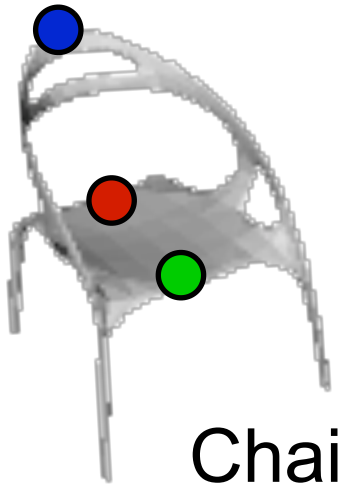


Pairwise Correspondences

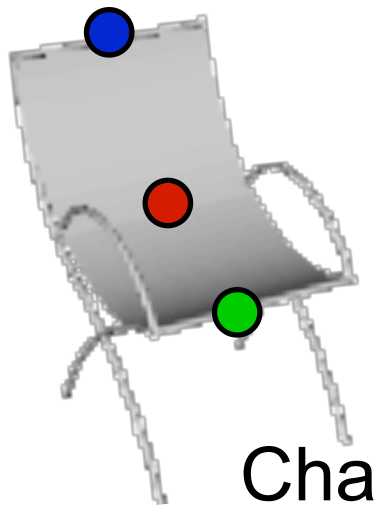
Fuzzy Correspondences: $e^{\frac{-D_t(p_i,p_j)^2}{\sigma(p_i)^2}}$



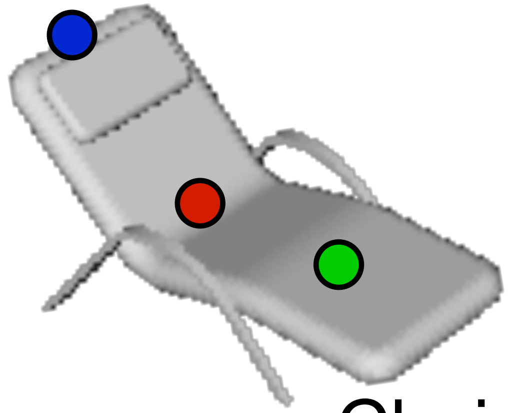
Chair 1



Chair 2

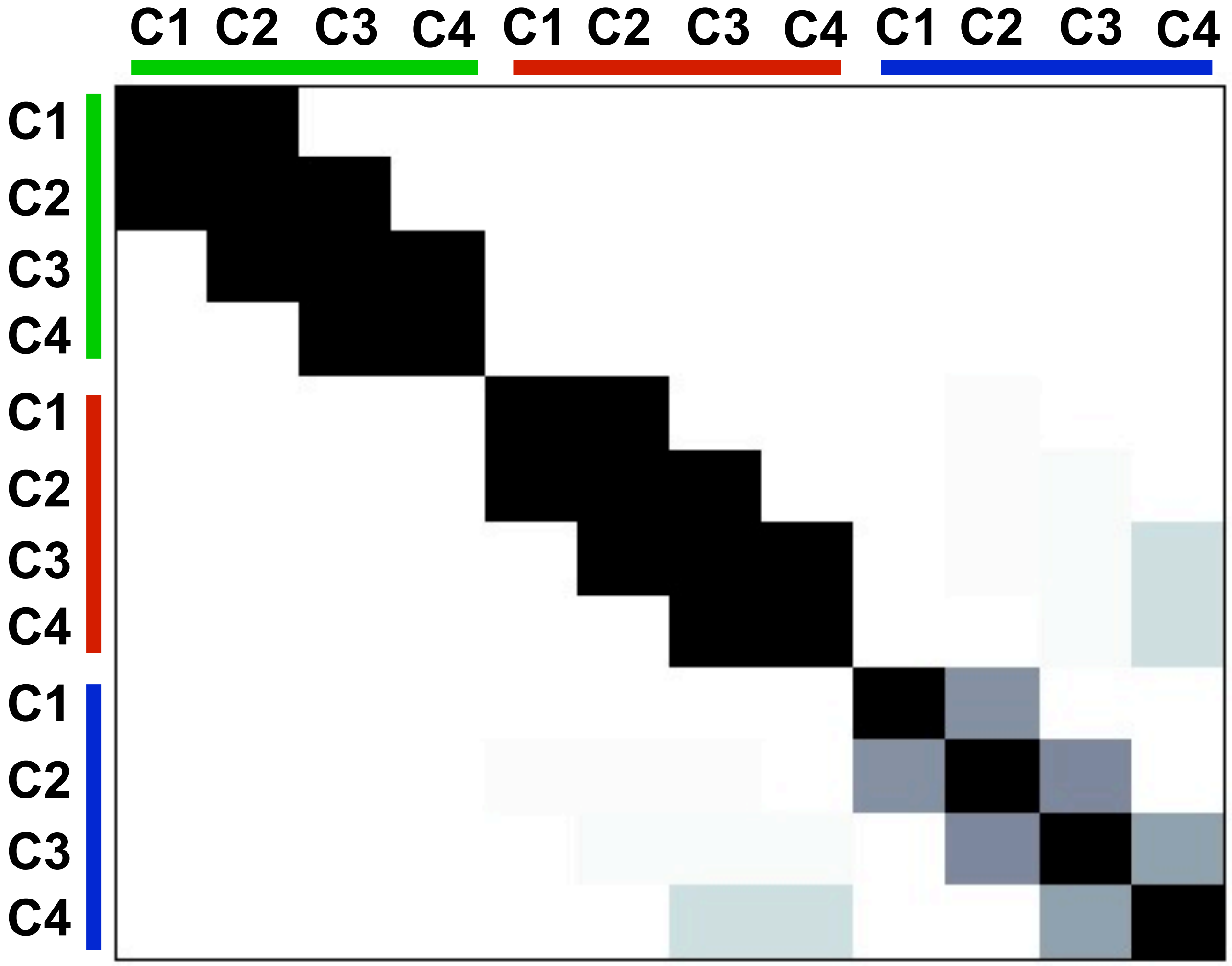


Chair 3

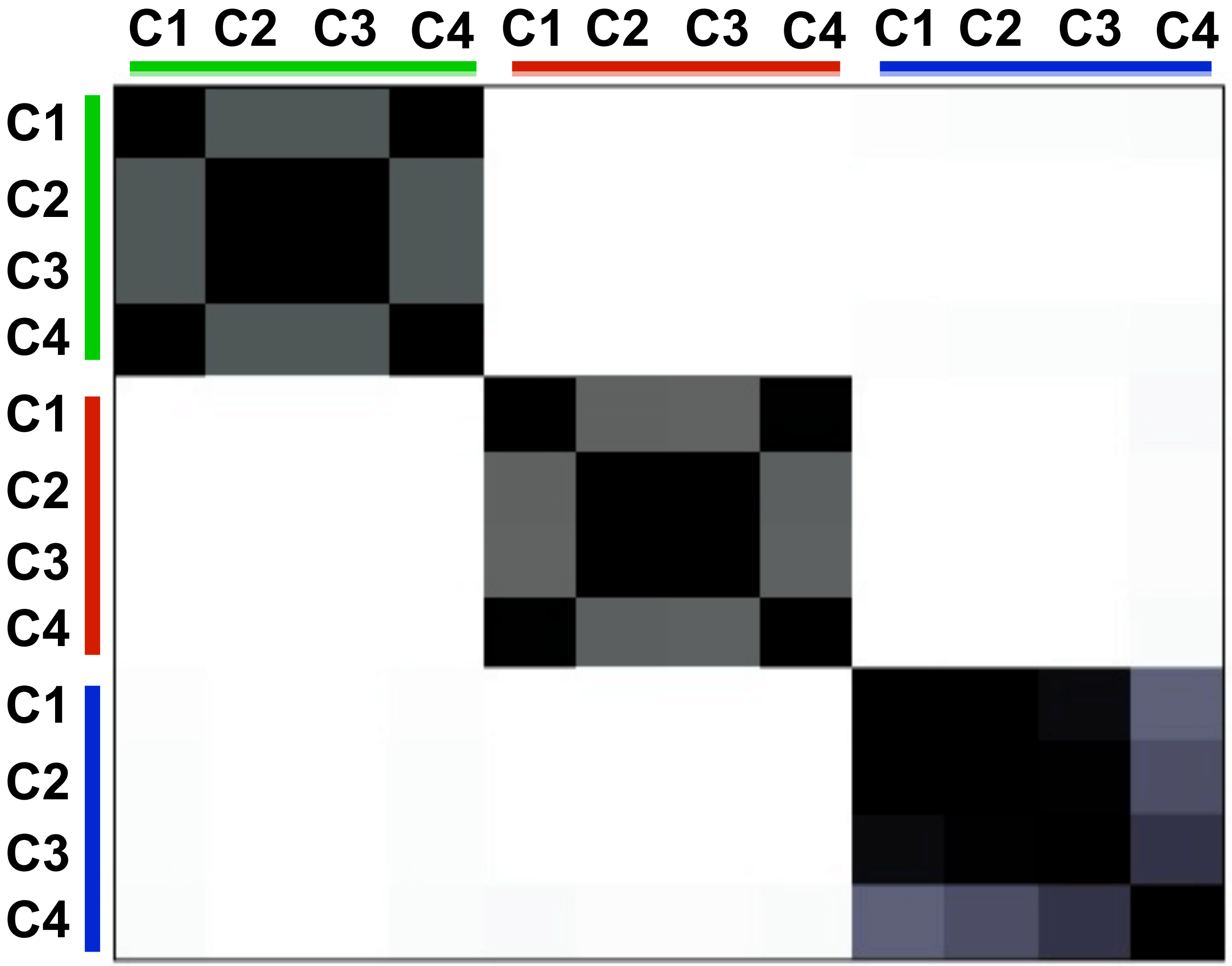


Chair 4

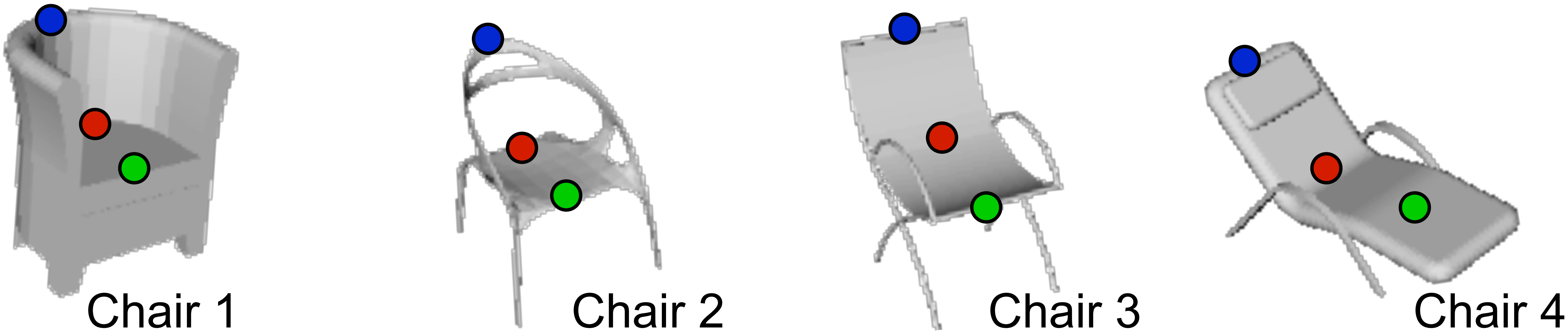
Diffusion Map



Pairwise Correspondences



Fuzzy Correspondences: $e^{\frac{-D_t(p_i,p_j)^2}{\sigma(p_i)^2}}$



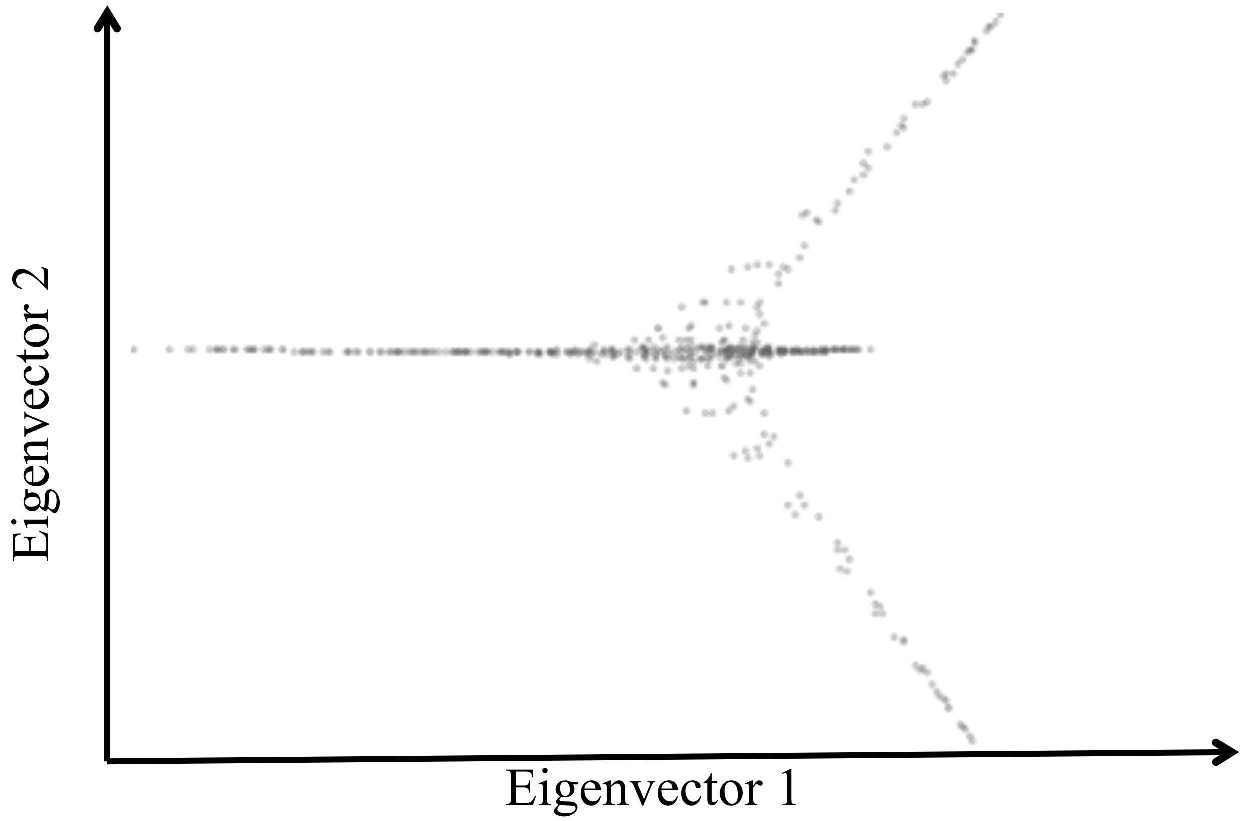
Larger embedding example

Embedding of 128 points from 7 planes

Larger embedding example

Embedding of 128 points from 7 planes

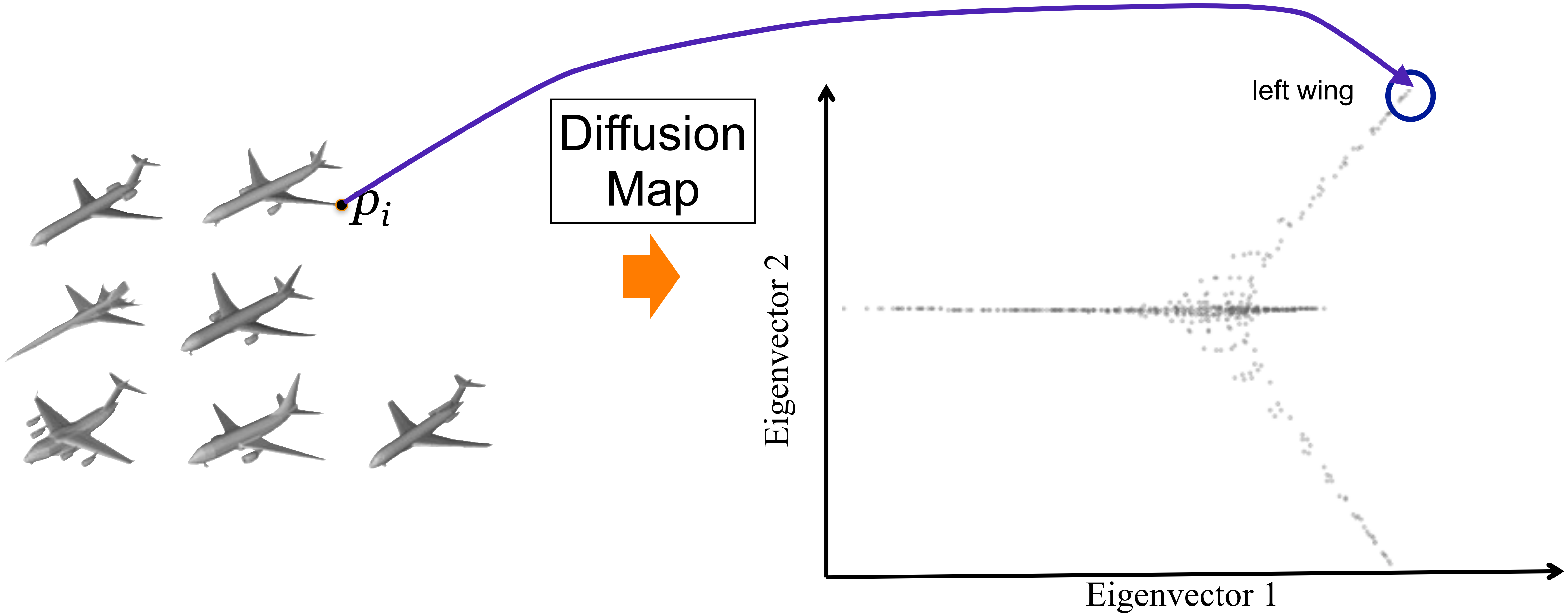
Diffusion Map



Spectral Embedding

Larger embedding example

Embedding of 128 points from 7 planes

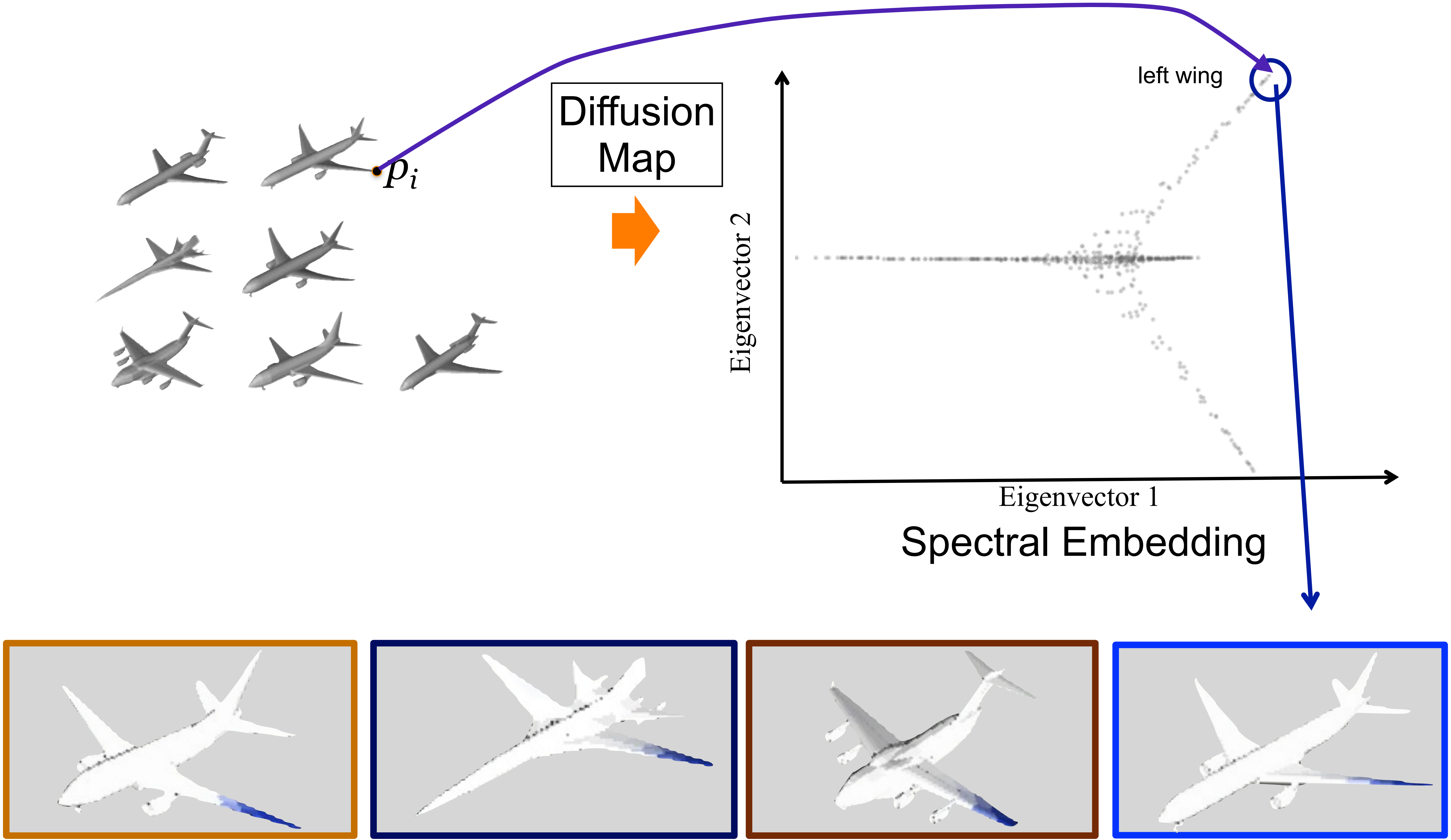


Diffusion Map

Spectral Embedding

Larger embedding example

Embedding of 128 points from 7 planes



p_i

Diffusion Map

left wing

Eigenvector 2

Eigenvector 1

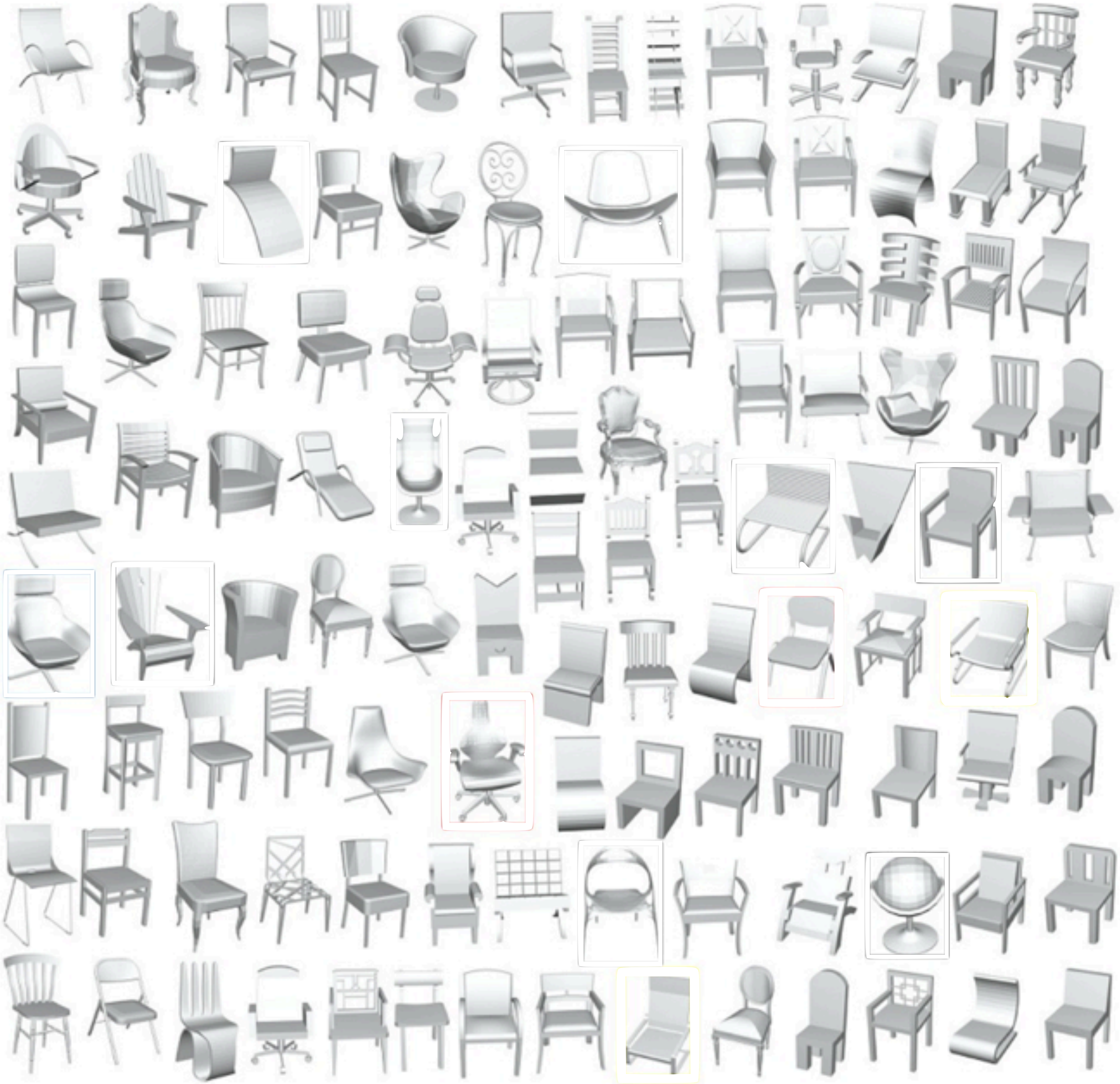
Spectral Embedding

Computing Fuzzy Correspondences

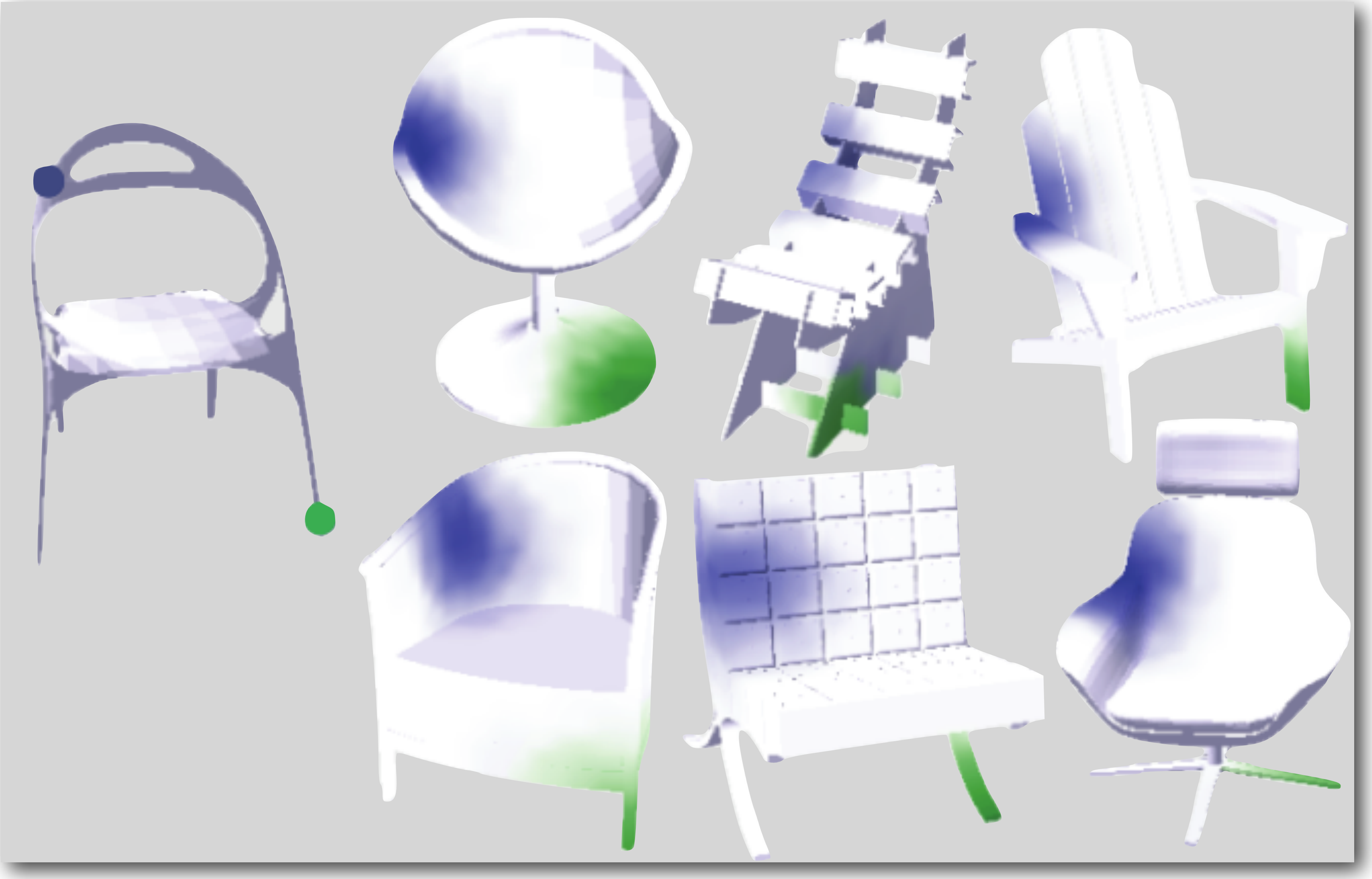
1. Sample points on each model
2. Select pairs of models to align
3. Estimate correspondences for selected pairs
4. Diffuse point correspondences
5. Re-align pairs to improve consistency

Repeat until convergence

Results

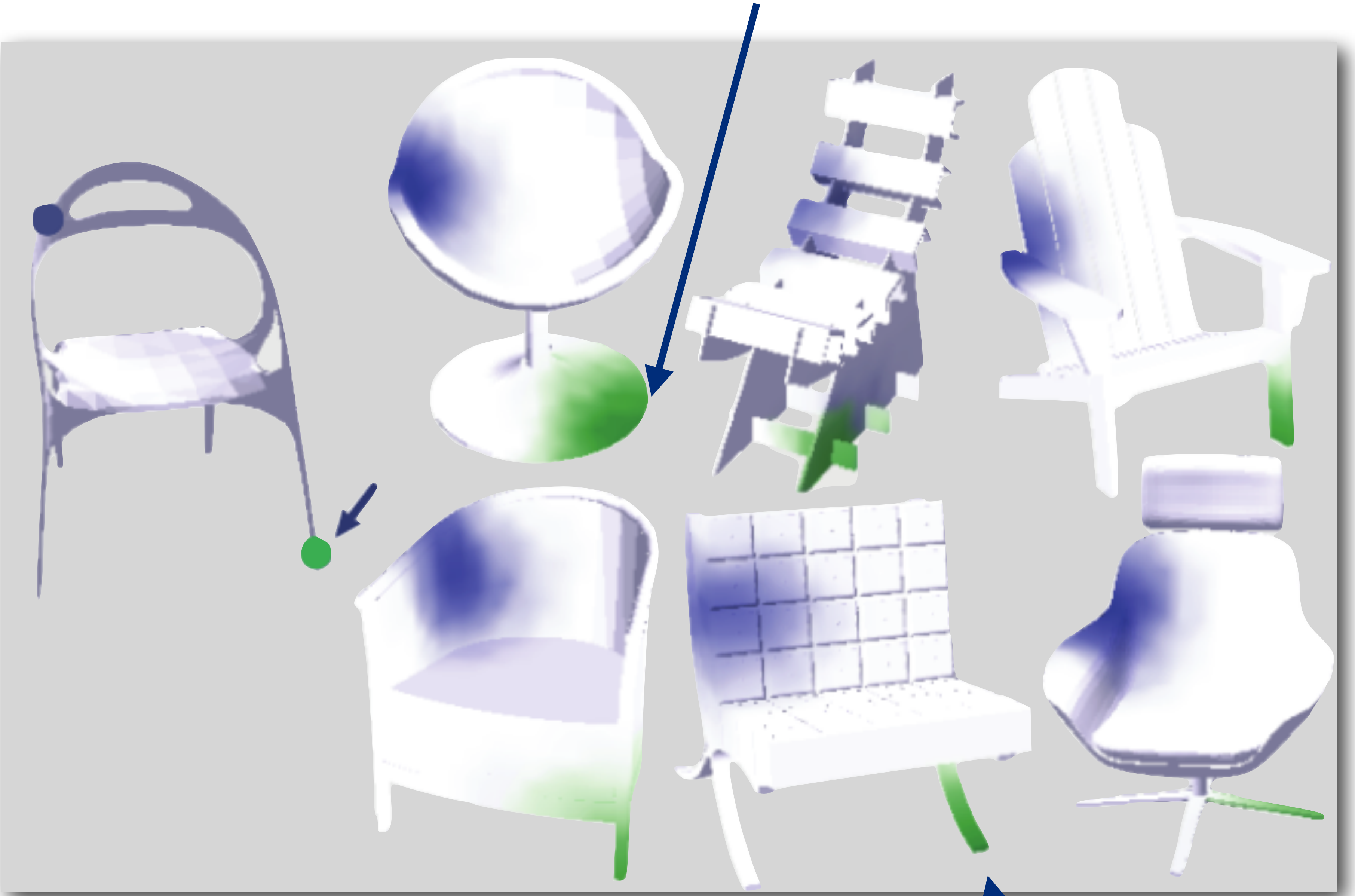


Results



Results

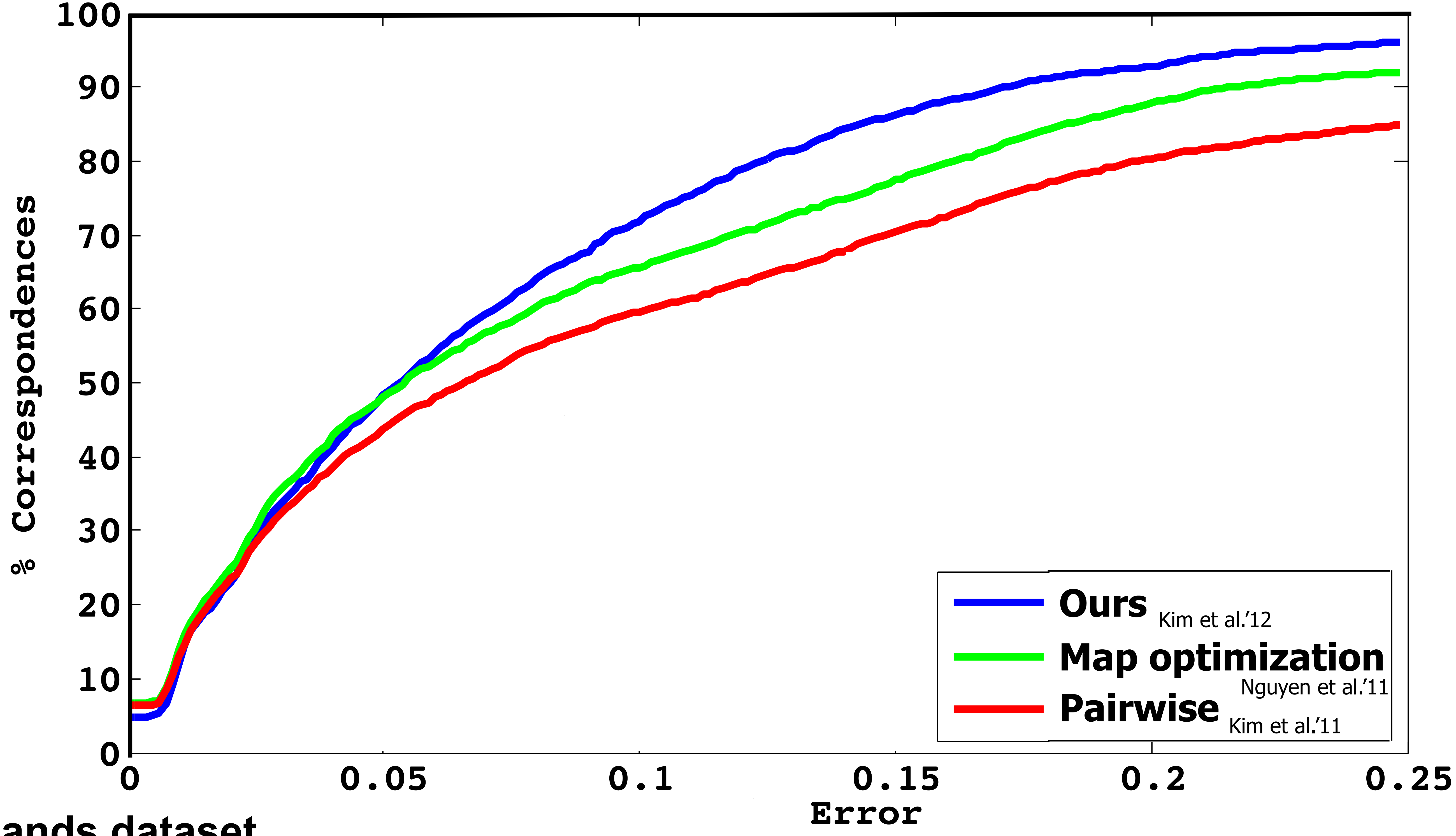
max=0.15



max=0.6

Results

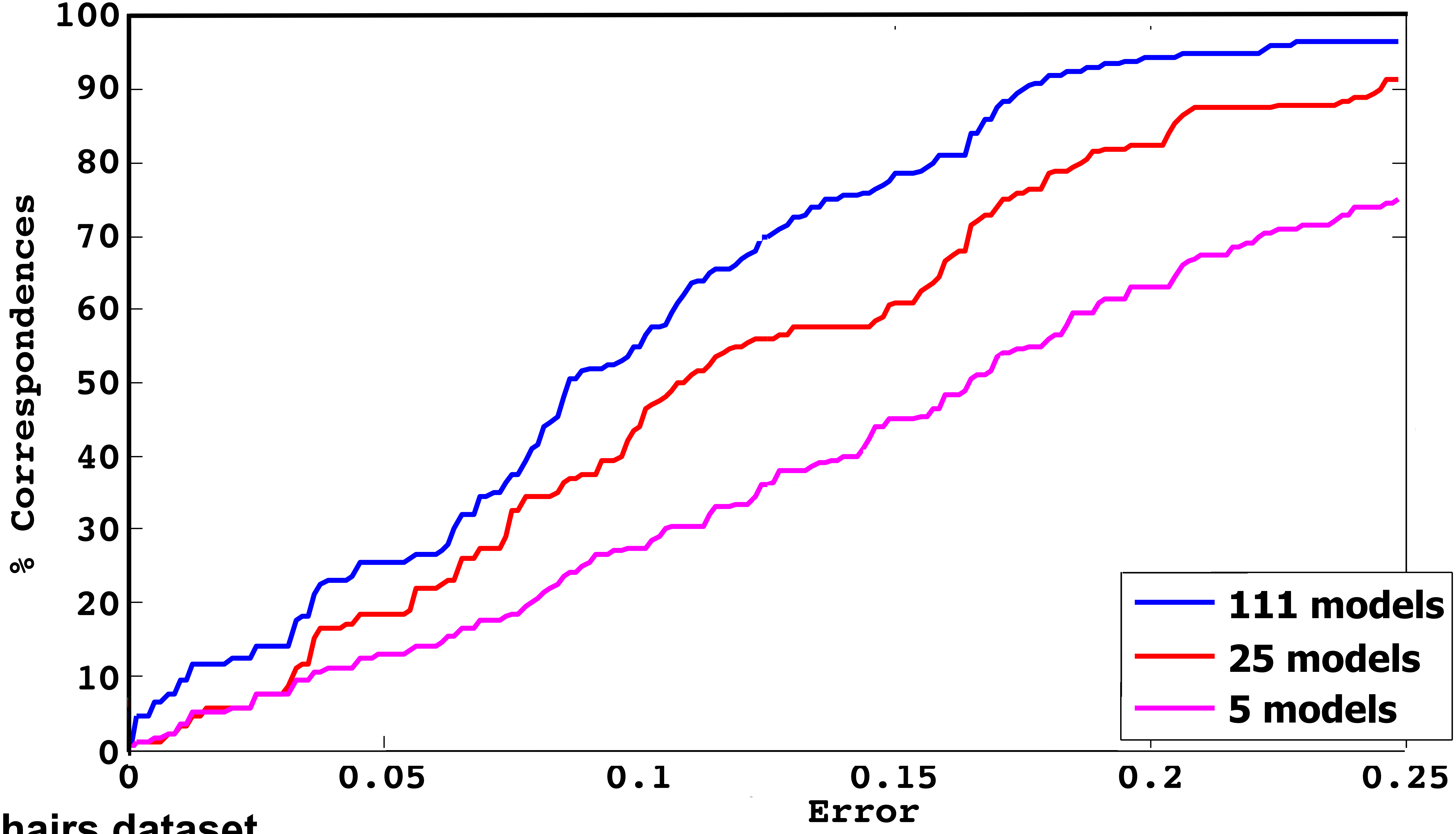
Our method compares favorably on benchmarks



Hands dataset

Results

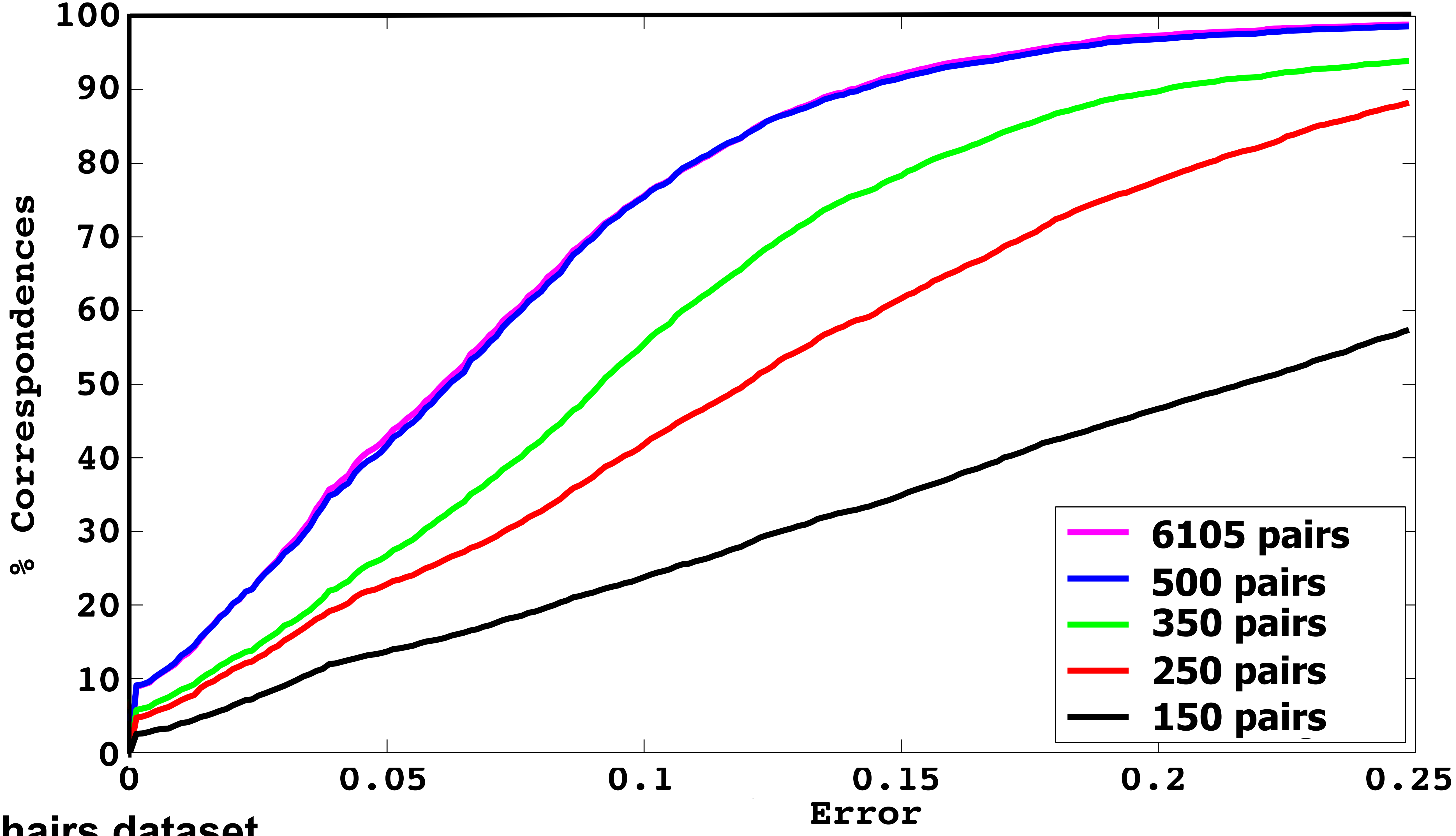
Larger collections yield better correspondences



Chairs dataset

Results

A small subset of pairwise alignments suffices



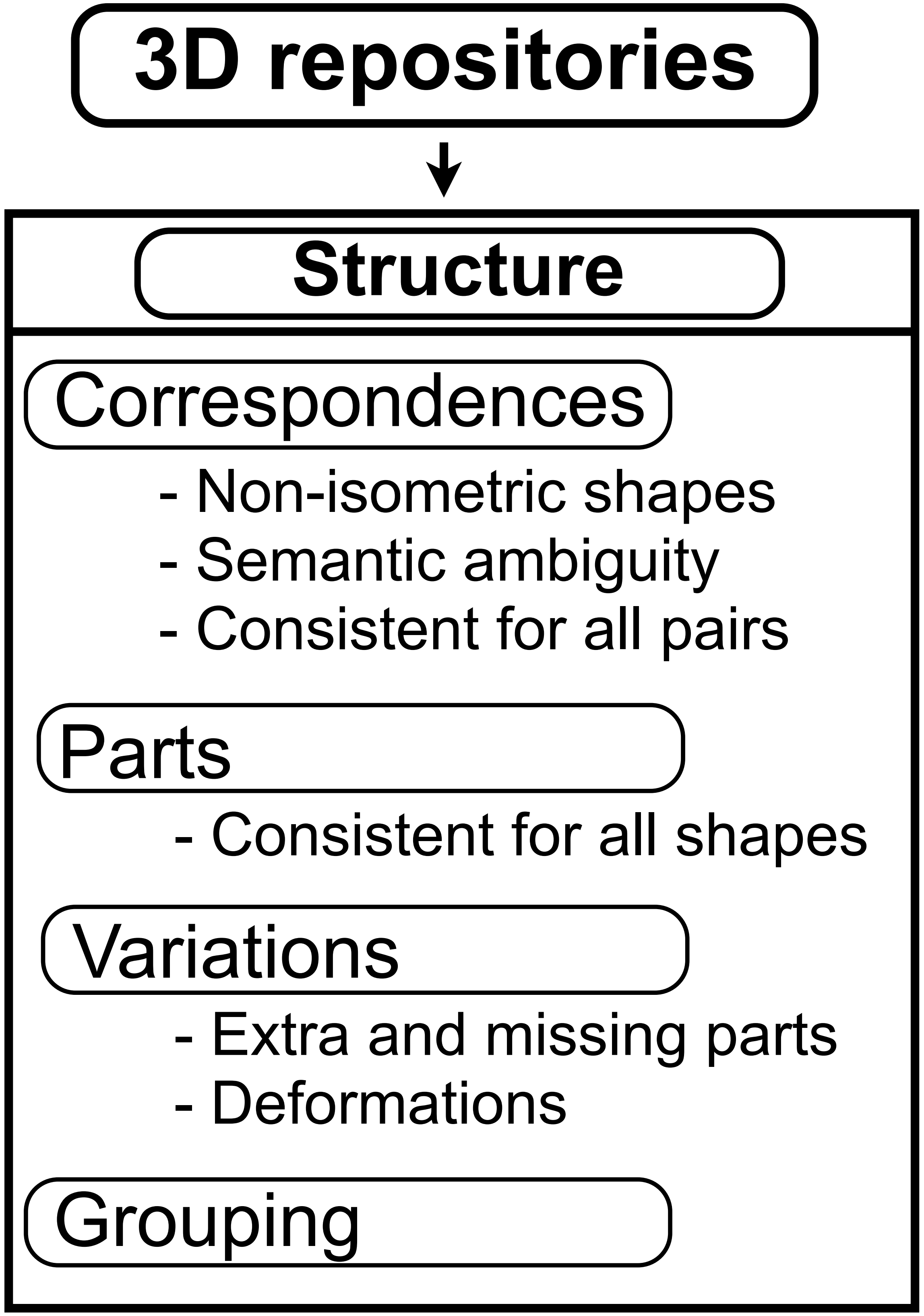
Chairs dataset

Summary

Fuzzy Correspondences via Diffusion

- Represent ambiguity in mapping
- Leverages transitivity to compare dissimilar shapes
- Far less than N^2 pairwise alignments are required

Talk Outline



3D repositories

Structure

Correspondences

- Non-isometric shapes
- Semantic ambiguity
- Consistent for all pairs

Parts

- Consistent for all shapes

Variations

- Extra and missing parts
- Deformations

Grouping

1. Blended Intrinsic Maps

2. Fuzzy Correspondences

3. Deformable Template

Complexity: O(N)

Goal

3D repositories

↓

Structure

Correspondences

- Non-isometric shapes
- Semantic ambiguity
- Consistent for all pairs

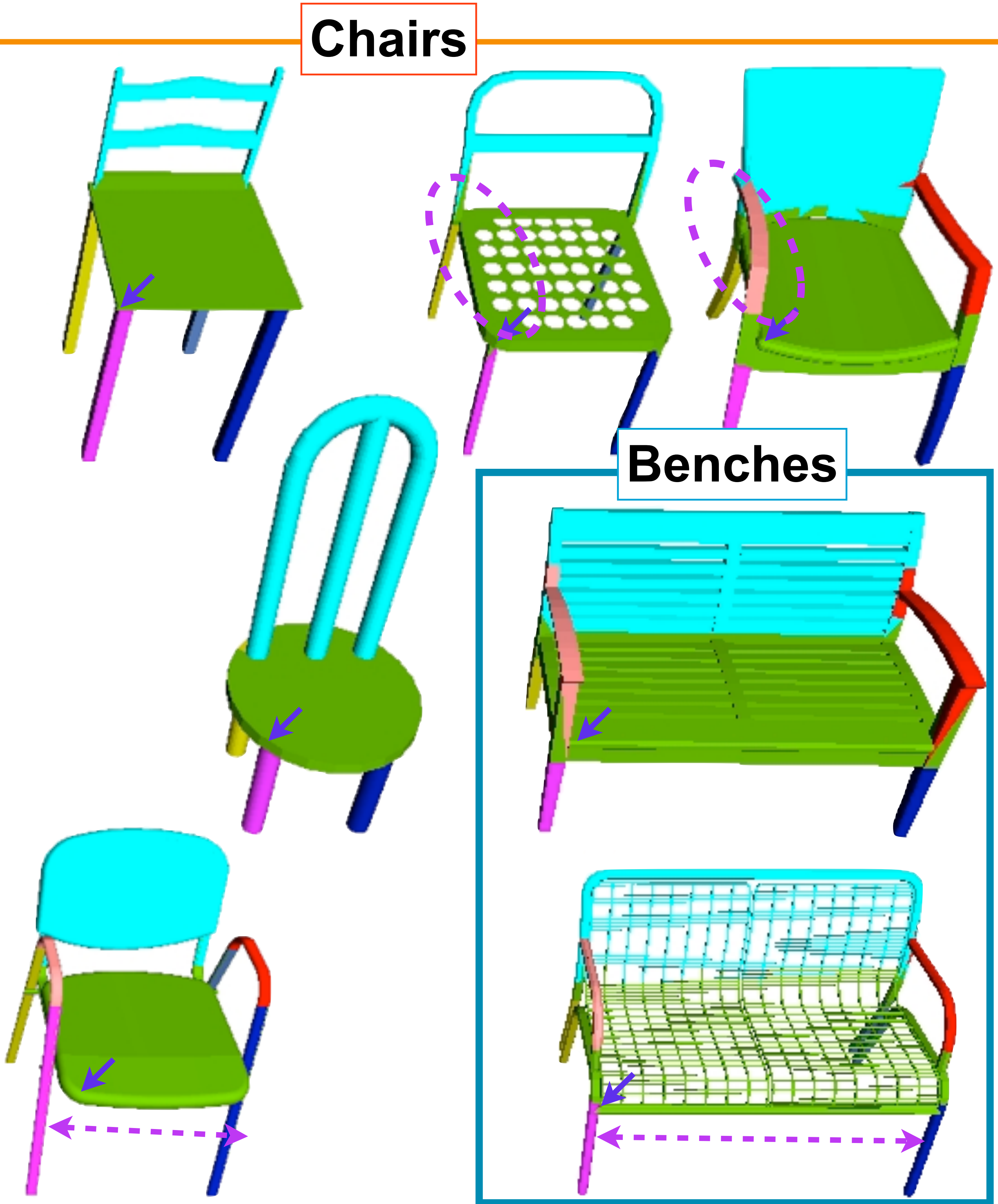
Parts

- Consistent for all shapes

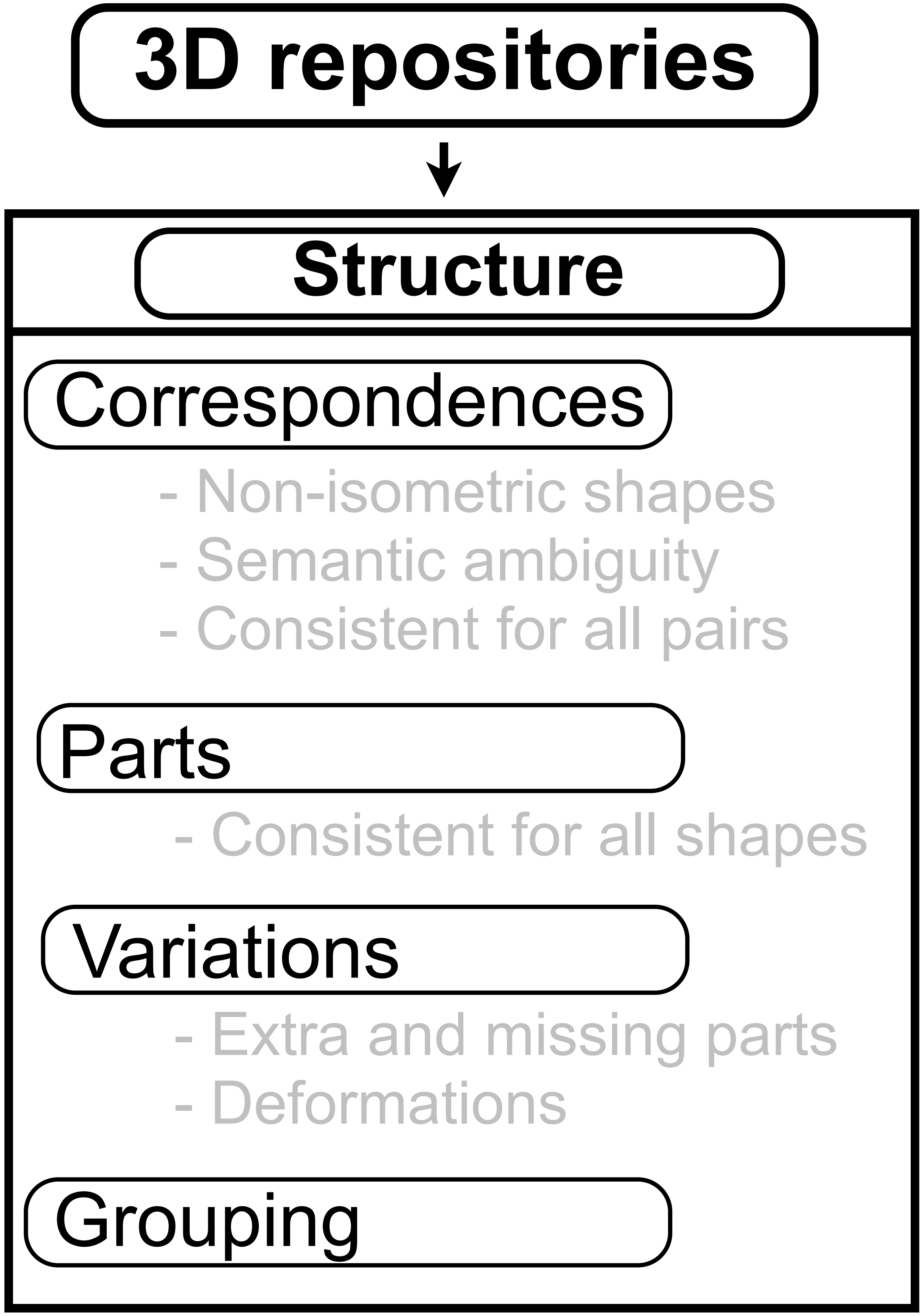
Variations

- Extra and missing parts
- Deformations

Grouping



Previous Work



3D repositories

Structure

Correspondences

- Non-isometric shapes
- Semantic ambiguity
- Consistent for all pairs

Parts

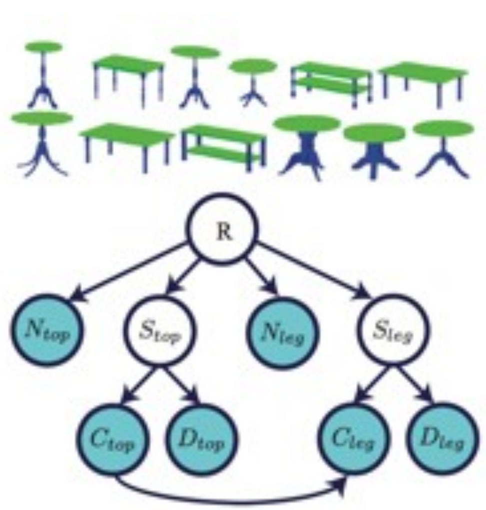
- Consistent for all shapes

Variations

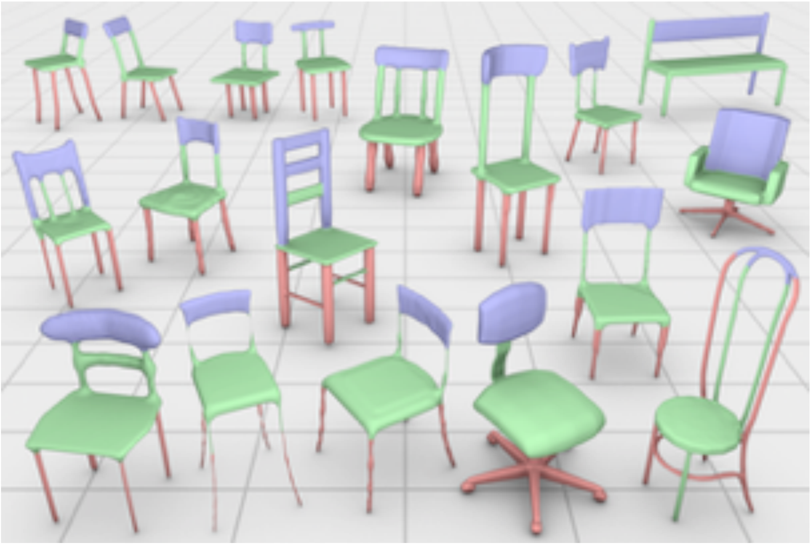
- Extra and missing parts
- Deformations

Grouping

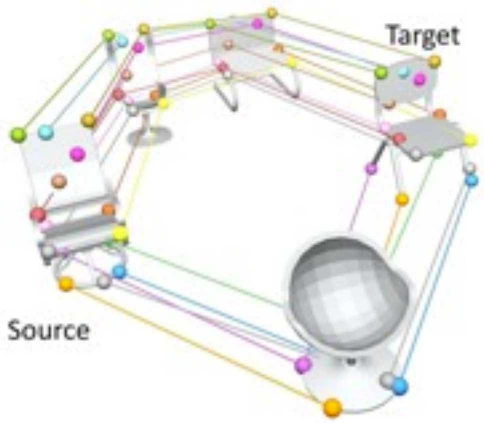
Each paper focuses only on one aspect



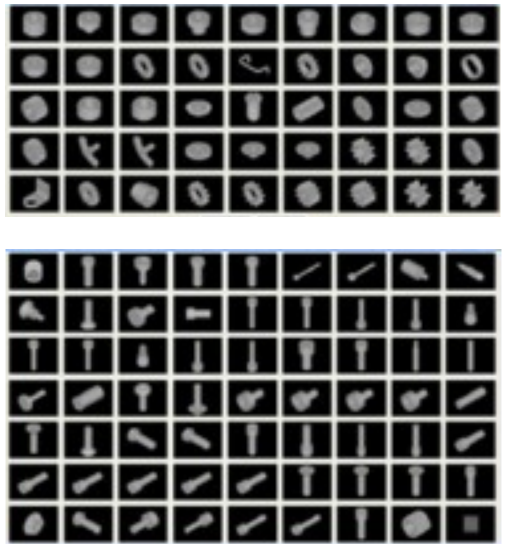
(c) Kalogerakis et al., SIGGRPAPH'12



(c) Sidi et al. SIGGRAPH Asia'11,
Huang et al. SIGGRAPH Asia'11



(c) Huang et al., SIGGRAPH Asia'12

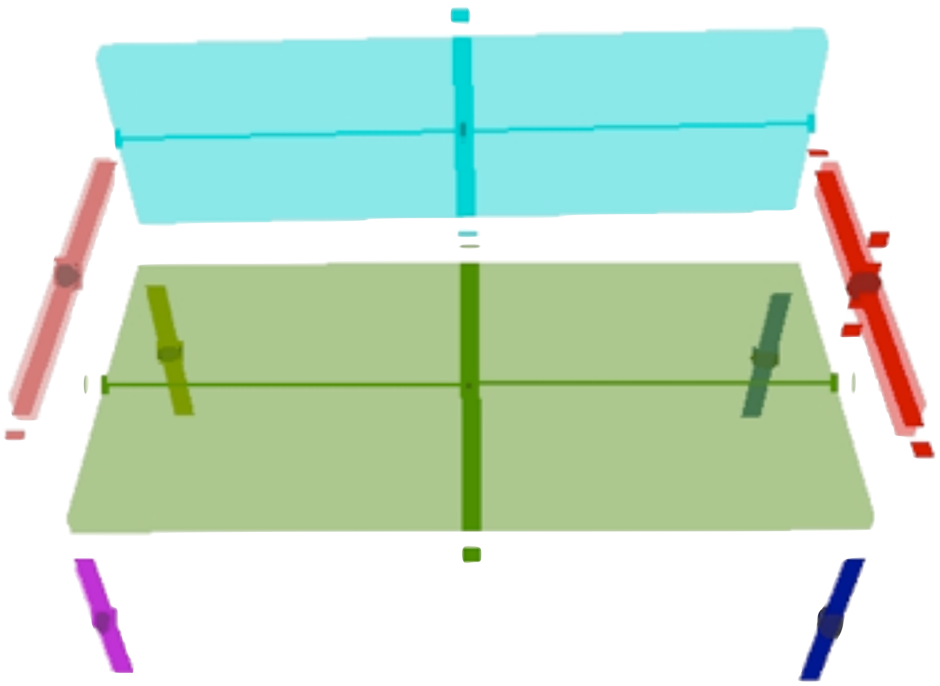


(c) Hou et al., CAD'05

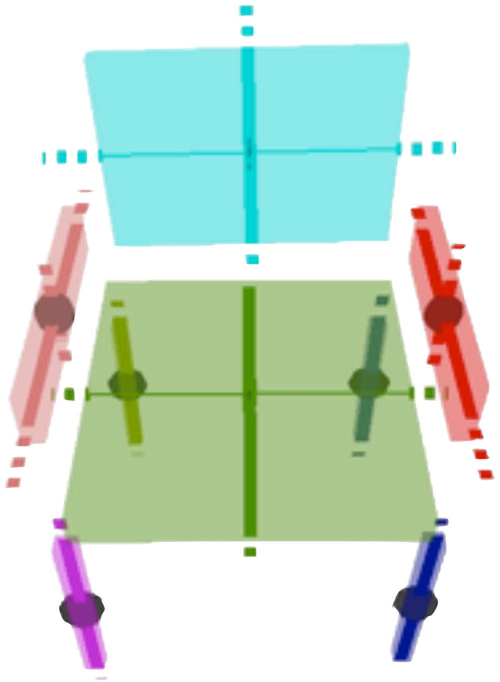
Our Approach

Learn part-based model with Gaussian distributions for

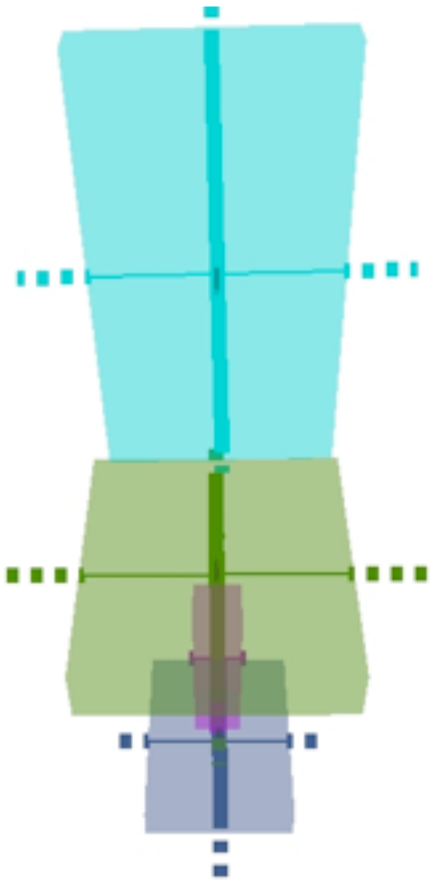
- Part positions
- Part anisotropic scales
- Part local shape features



Bench

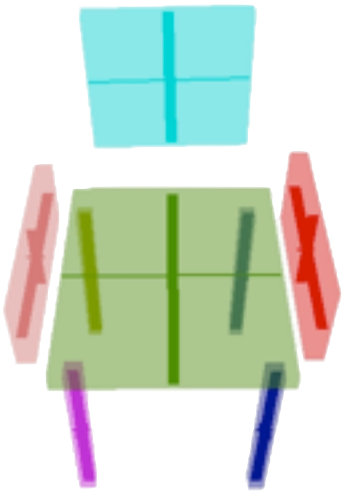


Dining chair

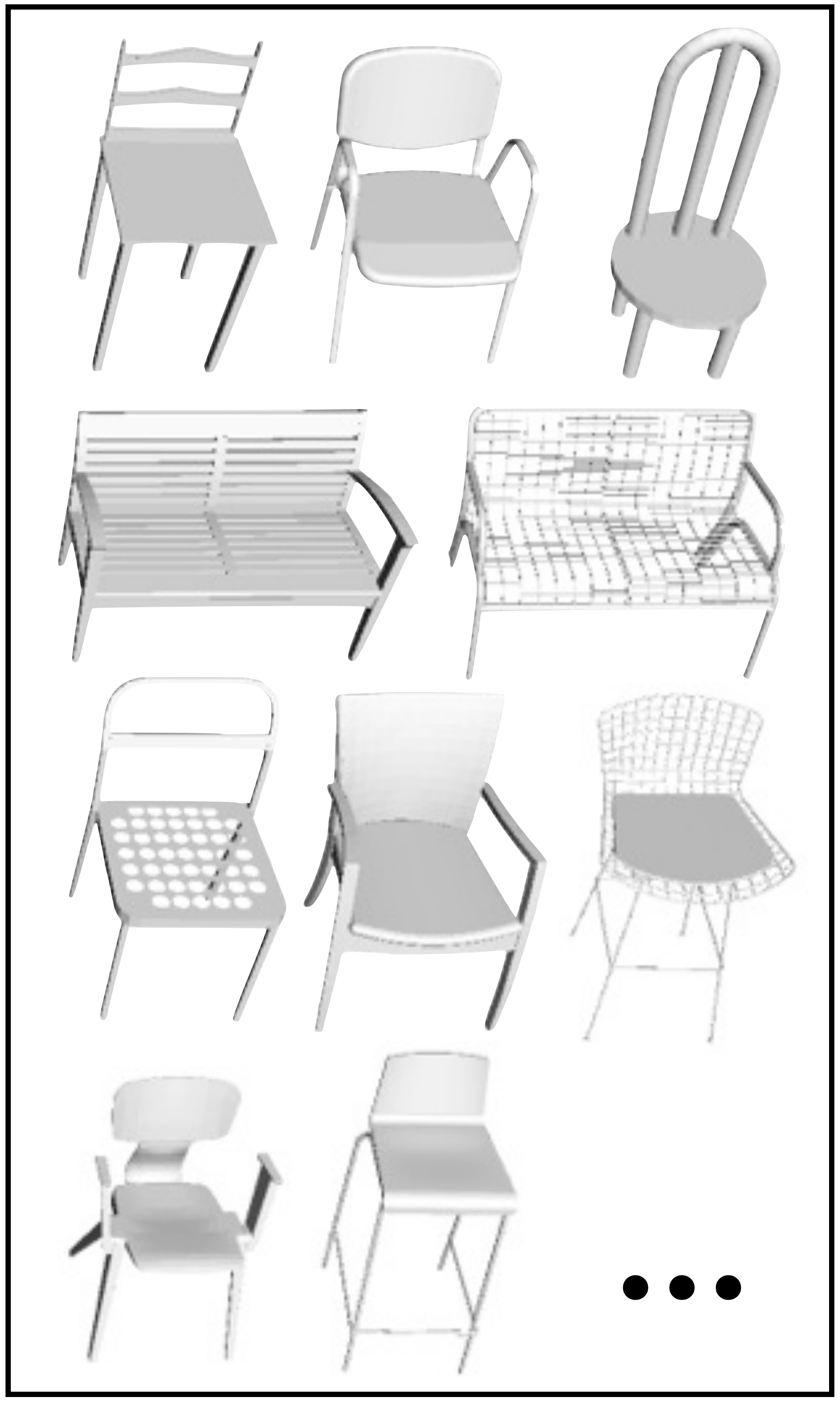


Swivel chair

Our Approach

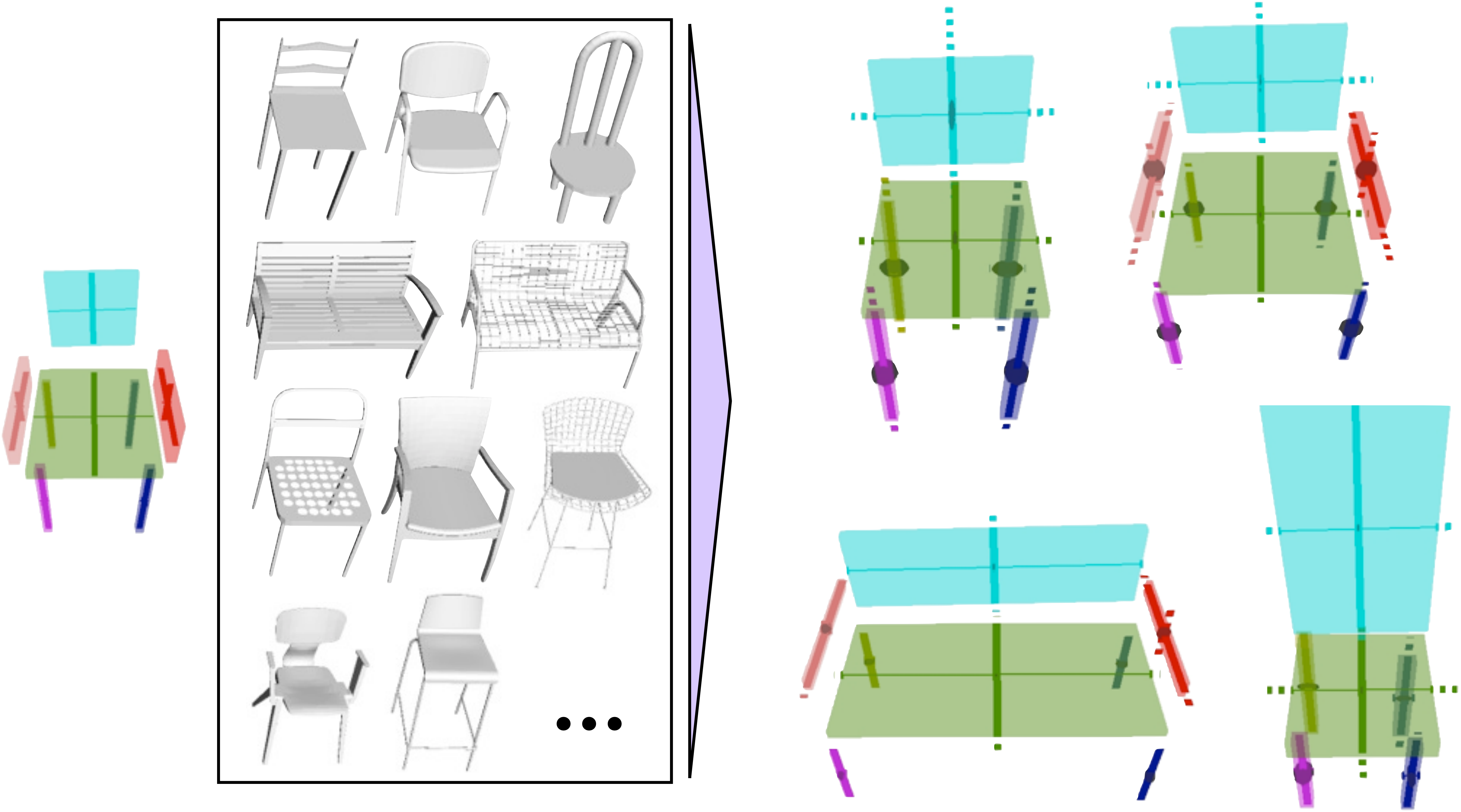


Initial template



Unlabeled, unorganized 3D collection

Our Approach

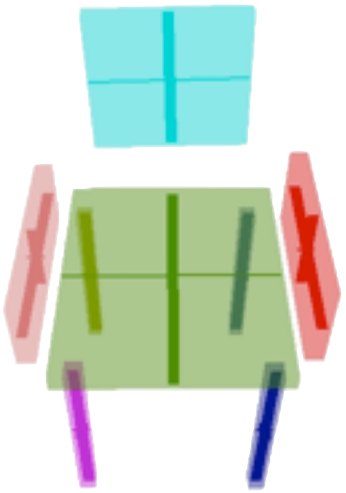


Initial template

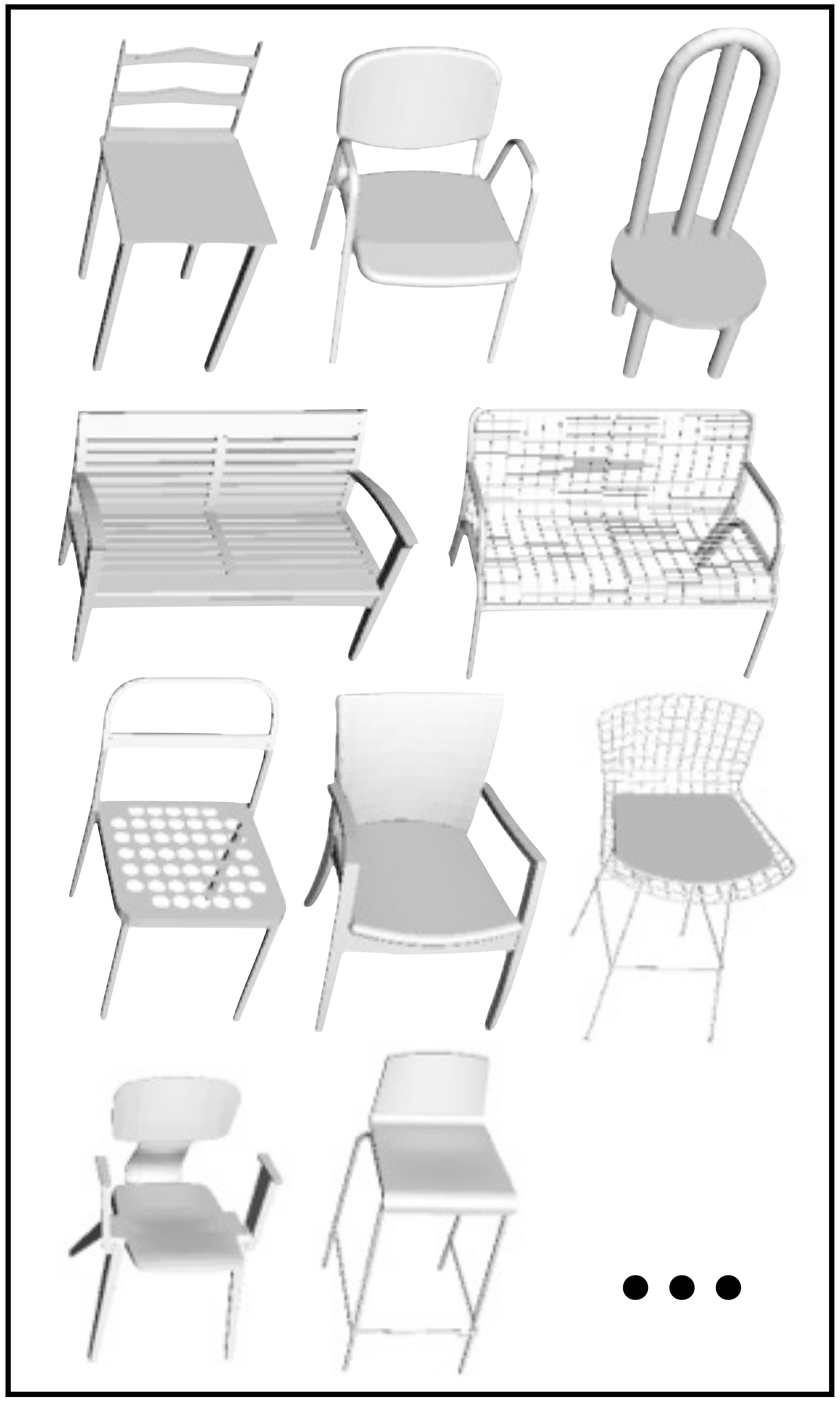
Unlabeled, unorganized 3D collection

Final Deformable Templates

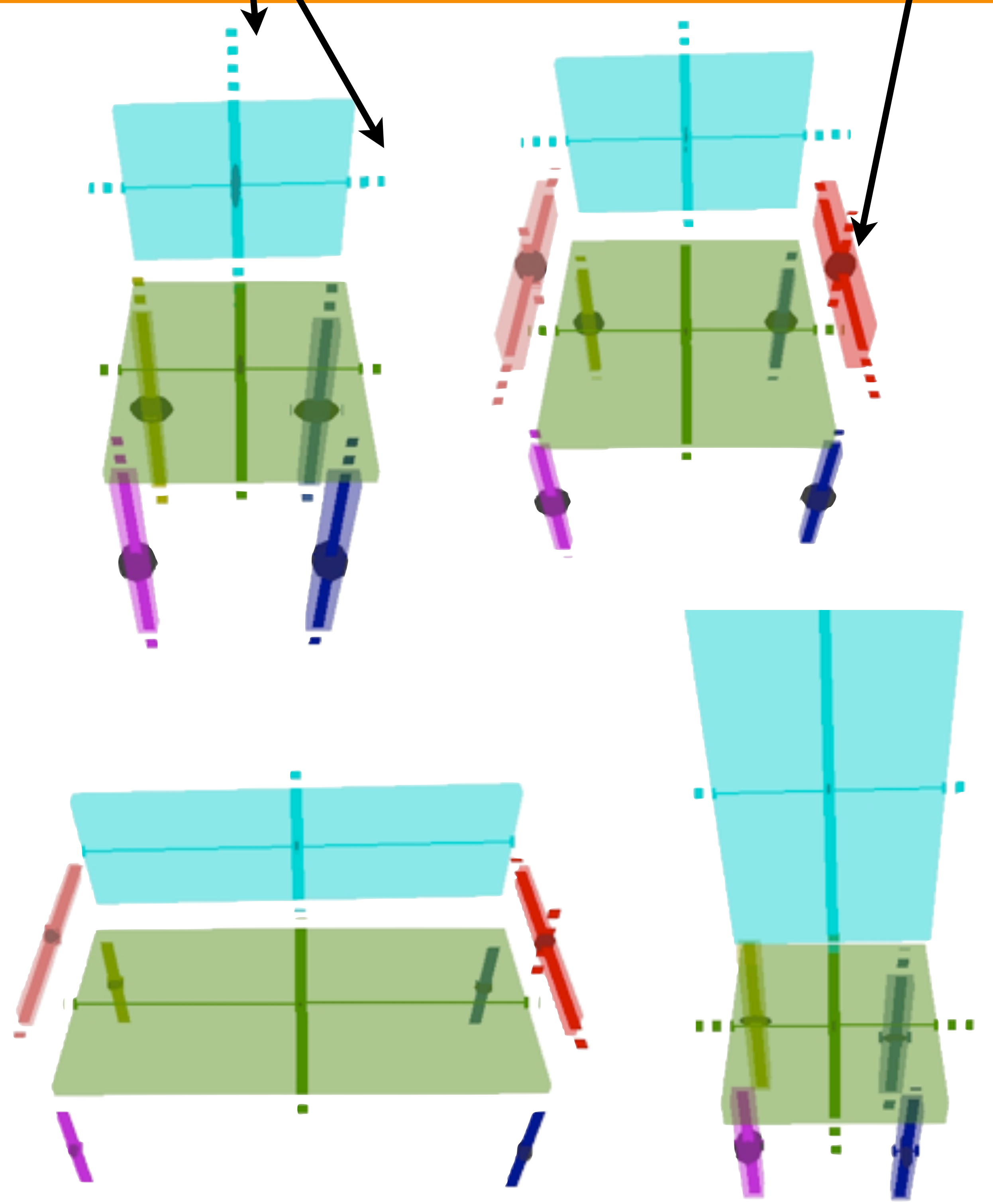
Our Approach



Initial template



Unlabeled, unorganized 3D collection



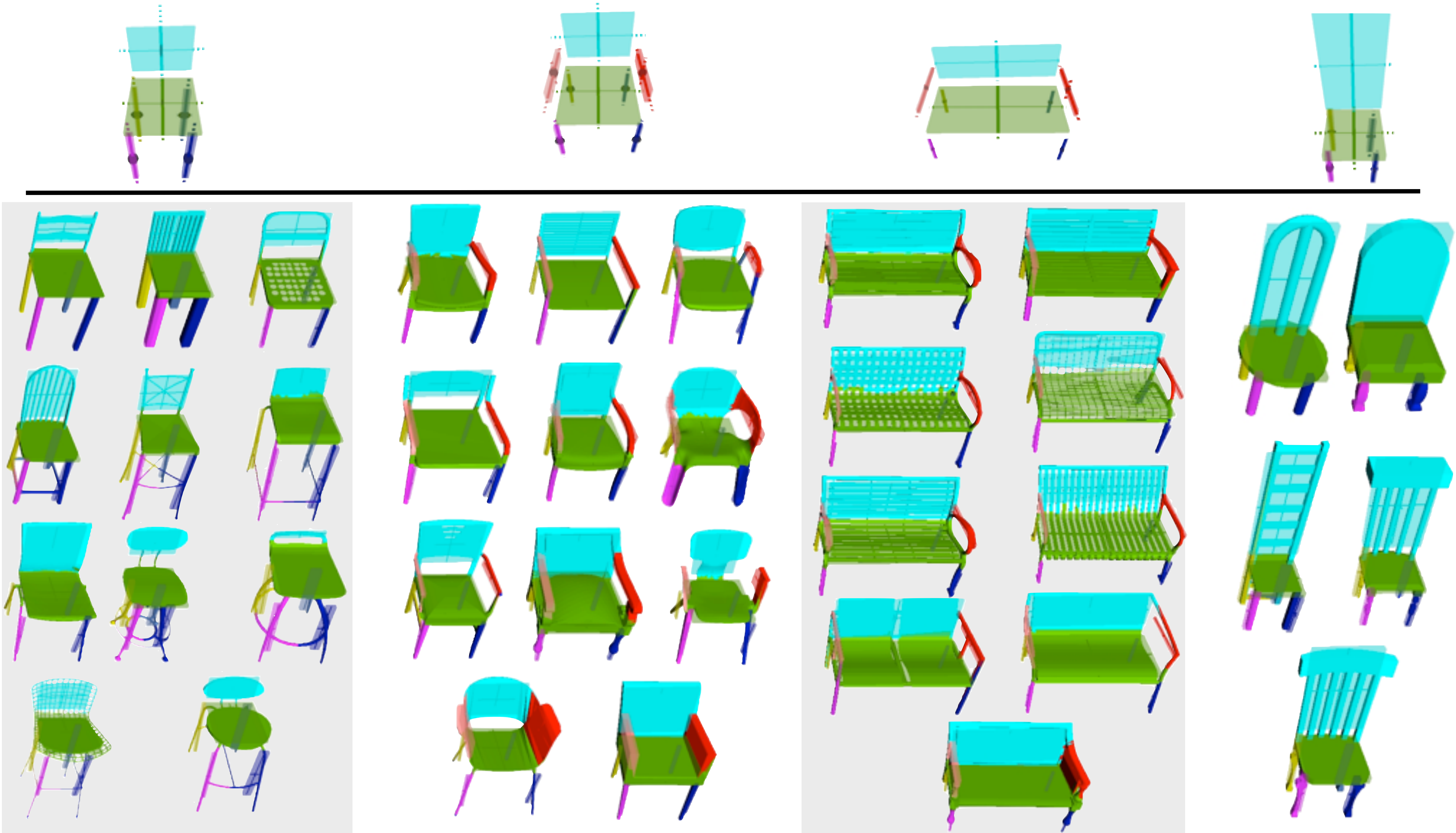
Back can get longer and wider

Arm rests vary in positions

Final Deformable Templates

Our Approach

Final Templates analysis results



Our Approach

Shape to template rigid alignment (r)

Per part deformations (d)

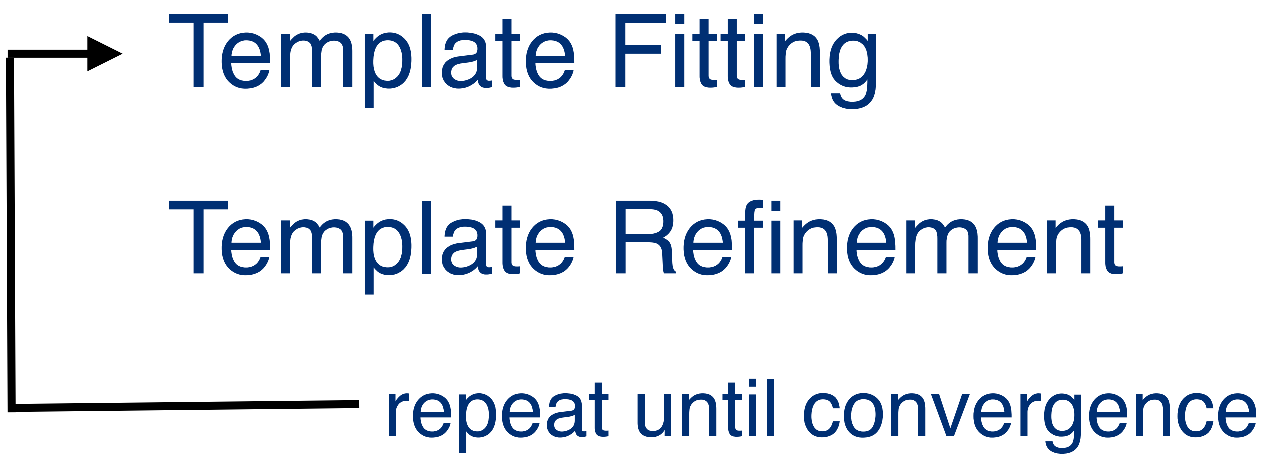
- Existence
- Centroid position
- Anisotropic scale

Labeling of points in the shape (ℓ)

Shape ↔ template mapping (m)

Method

Template Initialization

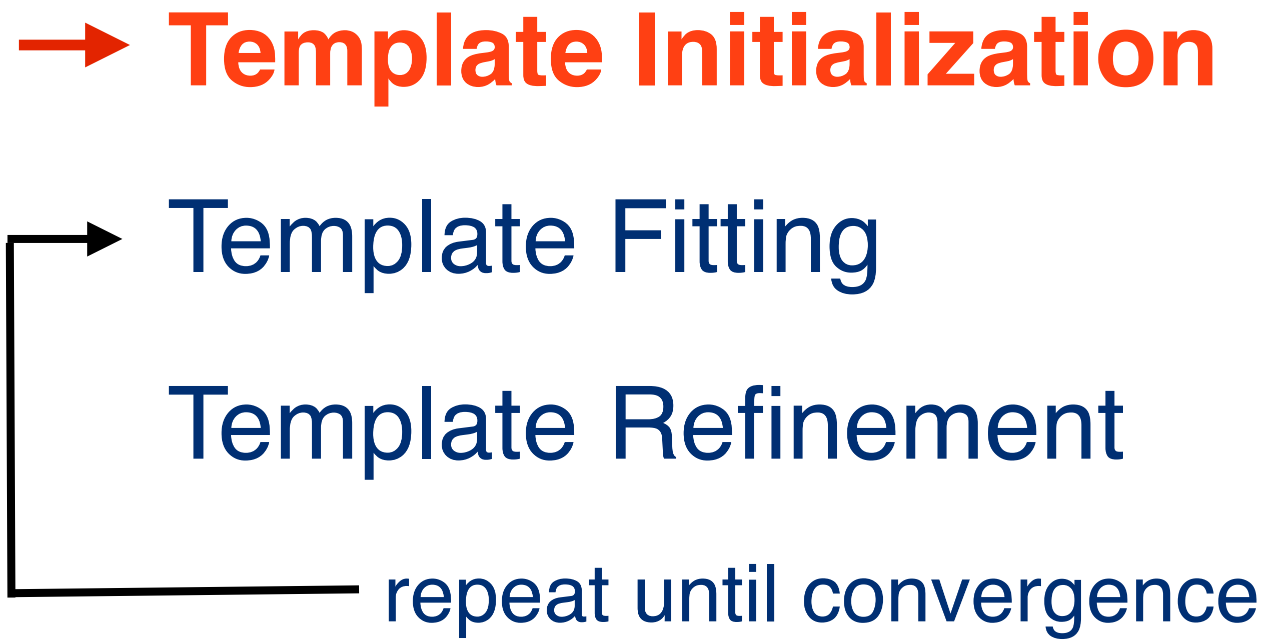


Template Fitting

Template Refinement

repeat until convergence

Method



Template Initialization

Template Fitting

Template Refinement

repeat until convergence

Template Initialization

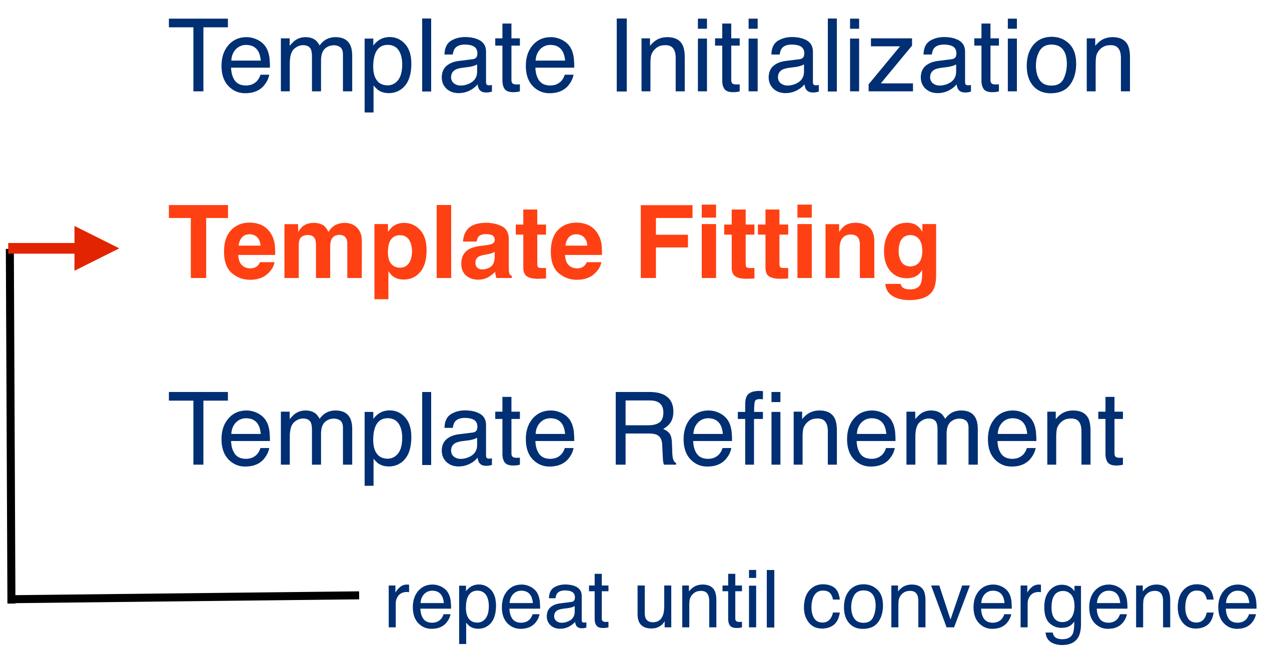
Manual initialization

- The user aligns boxes to semantic parts (≈ 5 min)

Automatic initialization

- Automatically segment all shapes
- Execute full template learning from best segmentations
- Pick template with smallest average fitting energy

Method



Template Initialization

Template Fitting

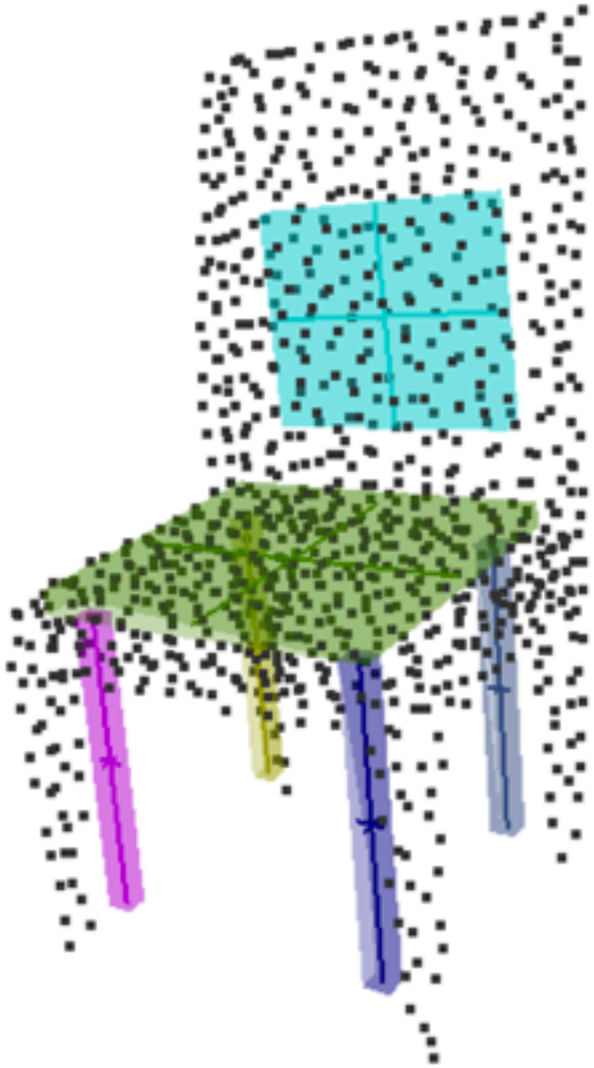
Template Refinement

repeat until convergence

Fitting Energy

$$E = E_{\mathrm{data}} + \gamma E_{\mathrm{deform}} + \beta E_{\mathrm{smooth}}$$

- E_{data} (template ⟷ shape distance + local shape features)
- E_{deform} (plausibility of template deformation)
- E_{smooth} (close & similar regions should get the same label)



Fitting Optimization

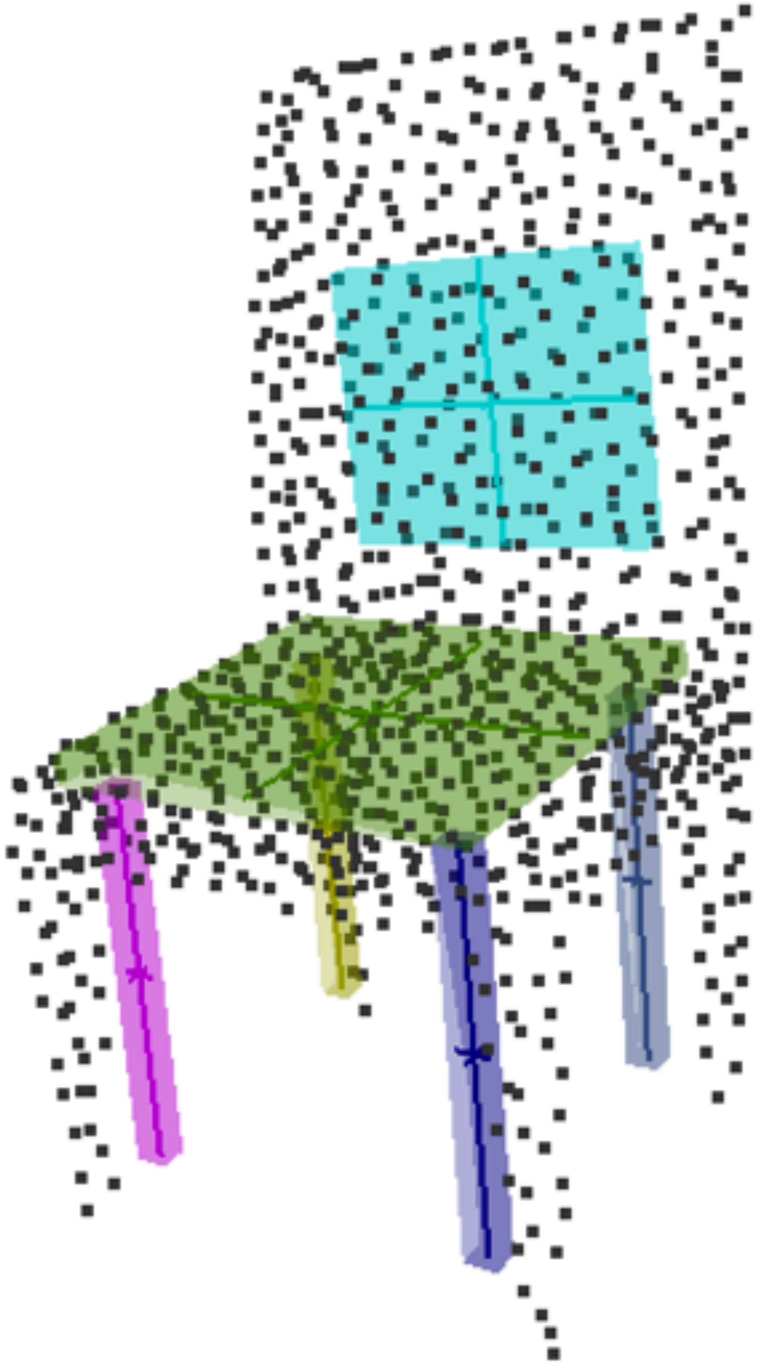
Alternate steps until shape segmentation converges:

- Segmentation (optimize ℓ)
- Correspondences (optimize m)
- Deformation (optimize r, d)

Fitting Optimization

Alternate steps until shape segmentation converges:

- Segmentation (optimize ℓ)
- Correspondences (optimize m)
- Deformation (optimize r, d)



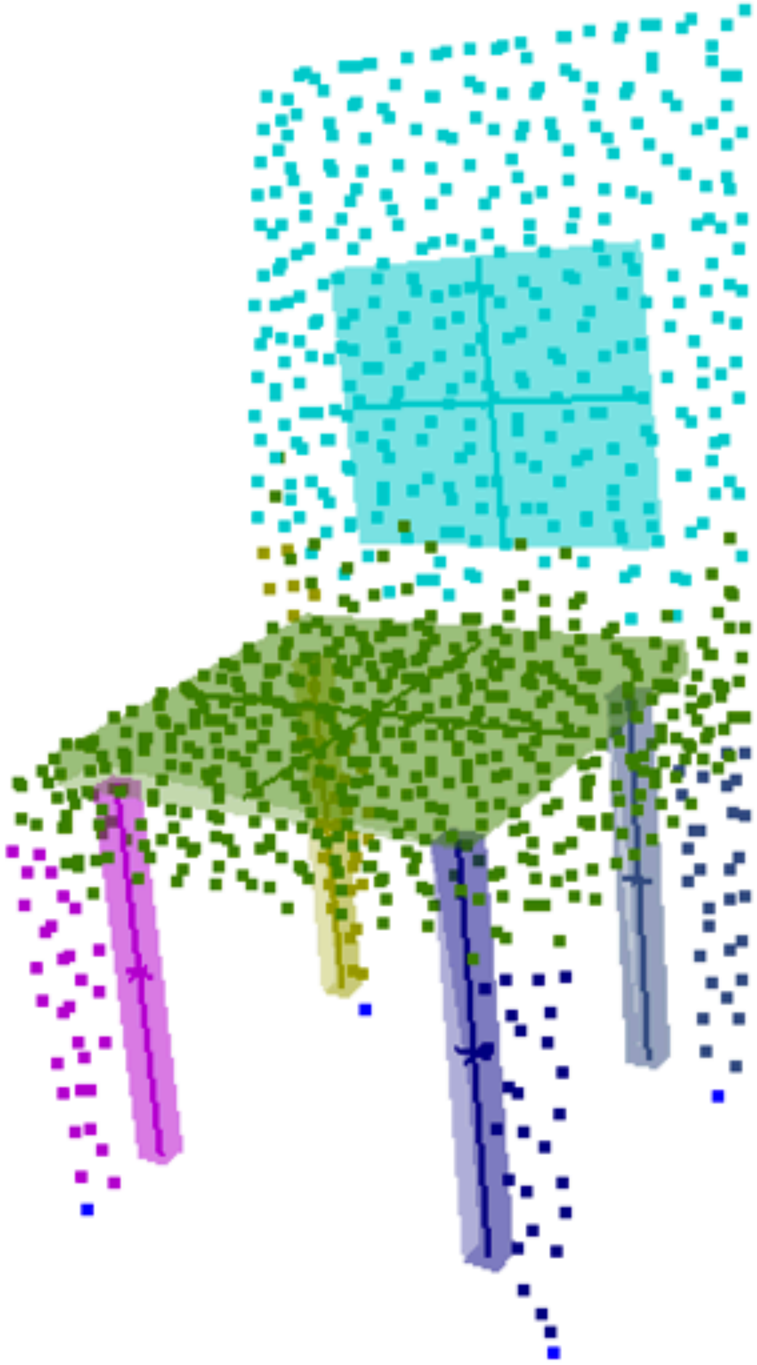
Fitting Optimization

Alternate steps until shape segmentation converges:

- ➡ Segmentation (optimize ℓ)
- ○ Correspondences (optimize m)
- ○ Deformation (optimize r, d)

$$E = \underline{E_{\text{data}}} + \gamma E_{\text{deform}} + \underline{\beta E_{\text{smooth}}}$$

Method: Graph cut [Boykov et al. 2001]



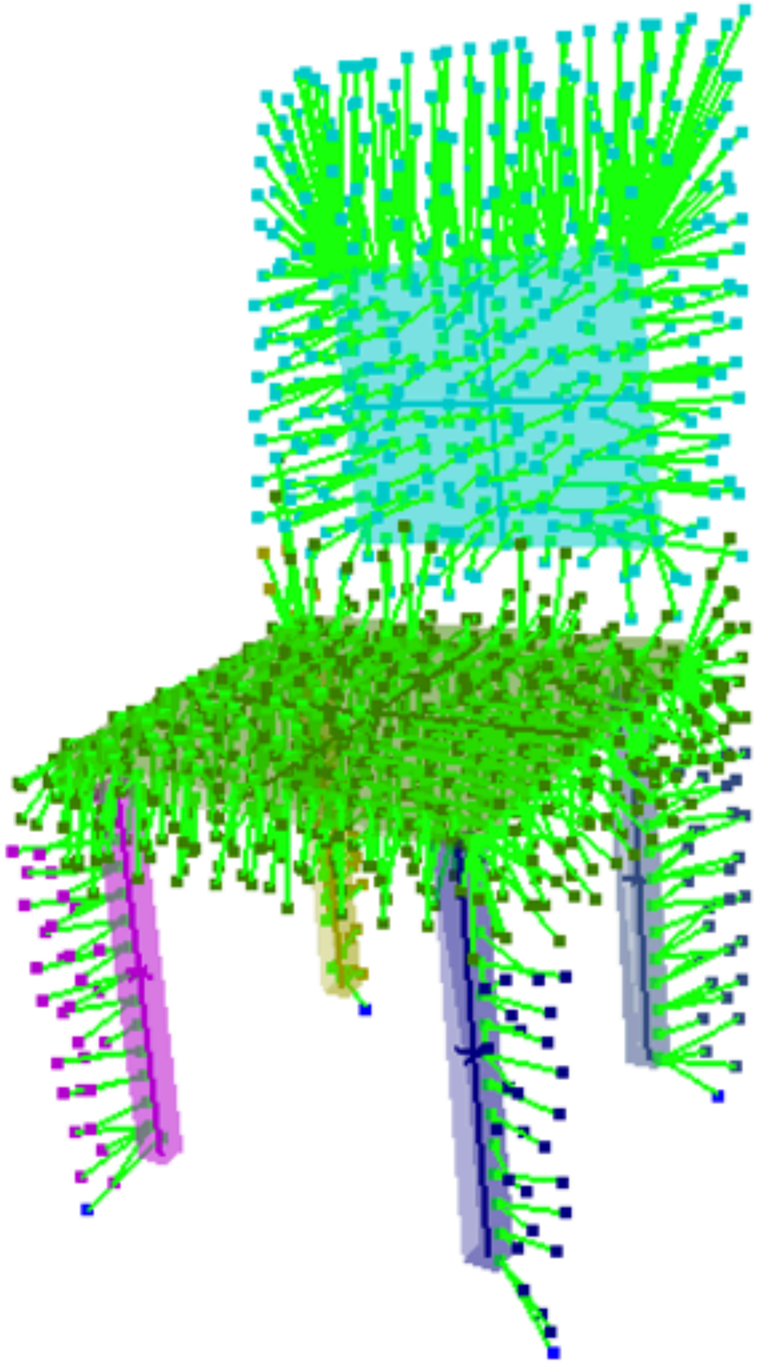
Fitting Optimization

Alternate steps until shape segmentation converges:

- Segmentation (optimize ℓ)
- ➡ Correspondences (optimize m)
- Deformation (optimize r, d)

$$E = \underline{E_{\text{data}}} + \gamma E_{\text{deform}} + \beta E_{\text{smooth}}$$

Method: Part-aware closest points



Fitting Optimization

Alternate steps until shape segmentation converges:

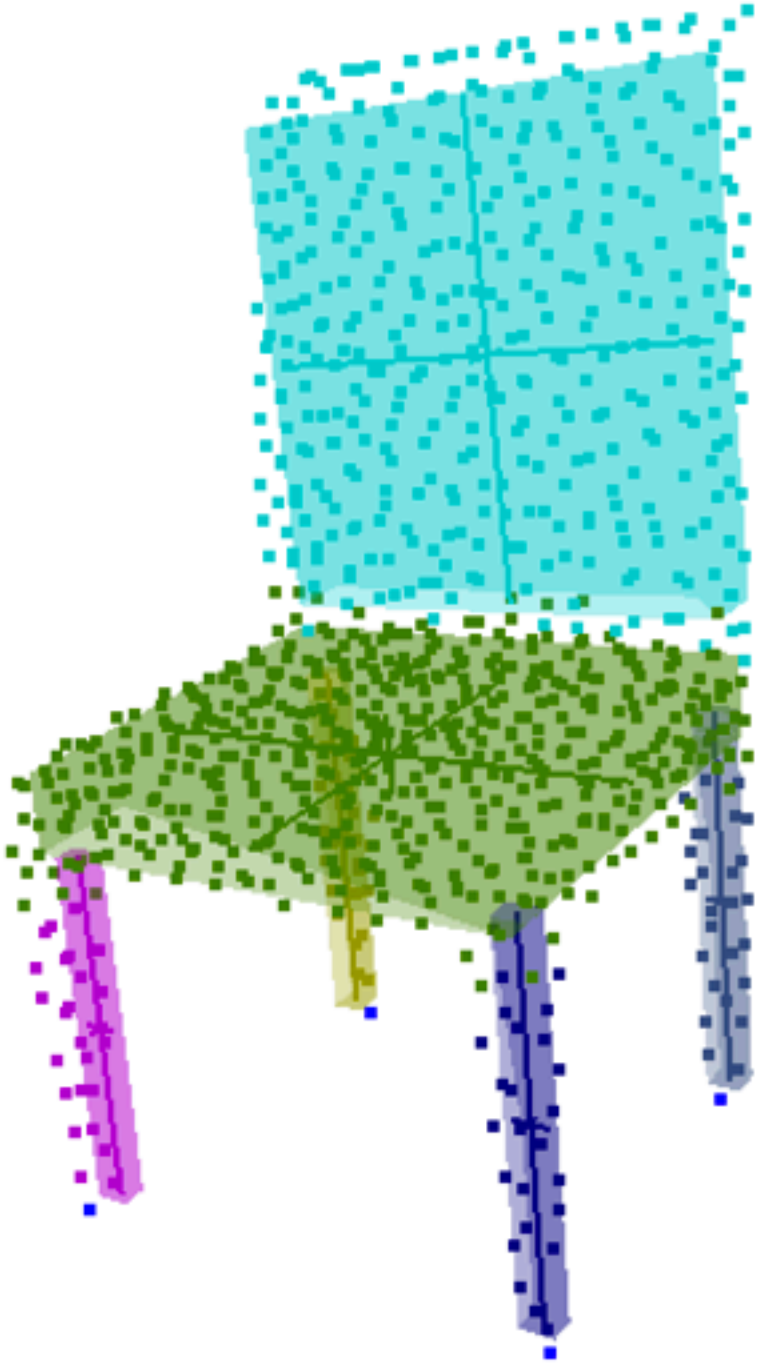
- Segmentation (optimize ℓ)
- Correspondences (optimize m)
- ➡ Deformation (optimize r, d)

$$E = \underline{E_{\text{data}}} + \gamma \underline{E_{\text{deform}}} + \beta E_{\text{smooth}}$$

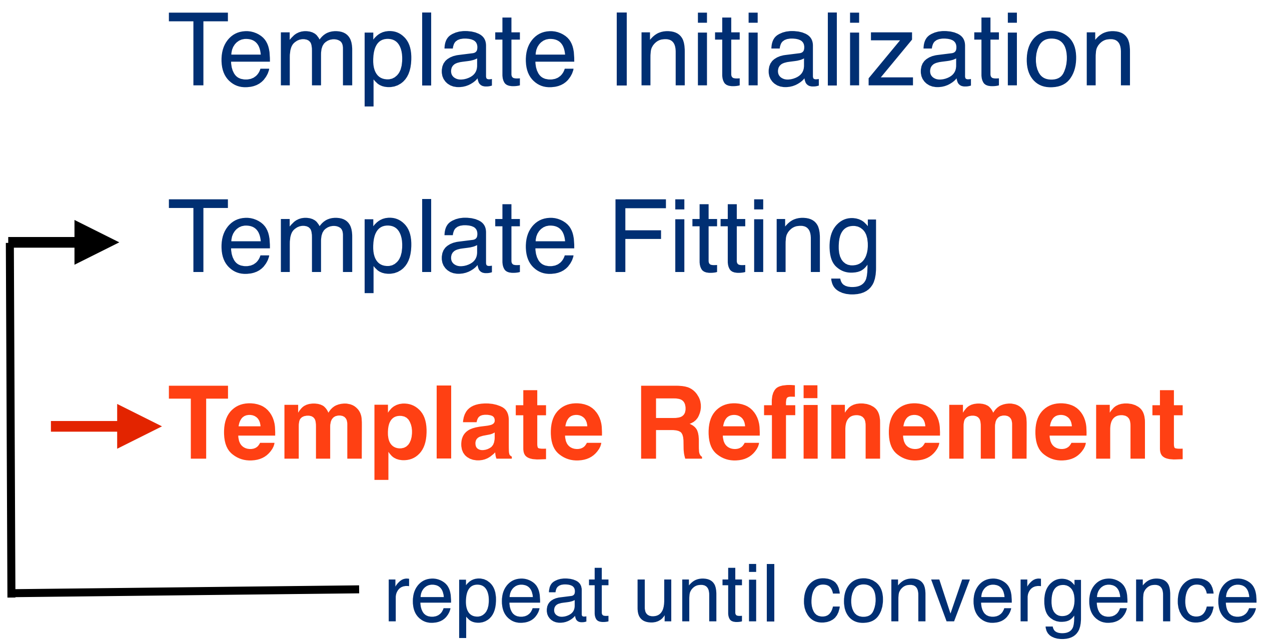
Method: Solve for critical points.

position: $$\frac{\partial(E_{\text{data}} + E_{\text{deform}})}{\partial b_p} = 0$$

scale: $$\frac{\partial(E_{\text{data}} + E_{\text{deform}})}{\partial b_s} = 0$$



Method



Template Initialization

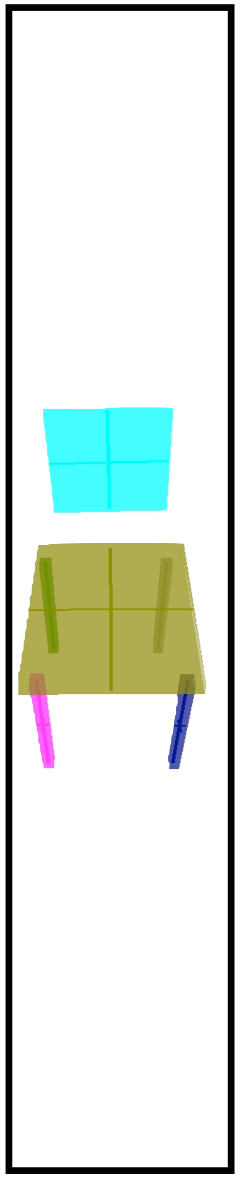
Template Fitting

Template Refinement

repeat until convergence

Overview

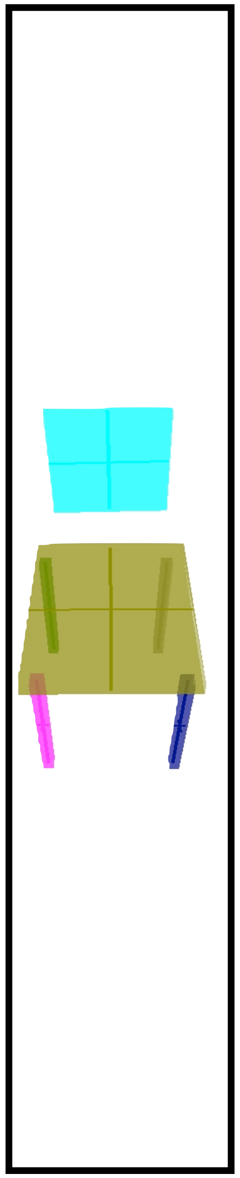
Improve set of templates from unlabeled geometry



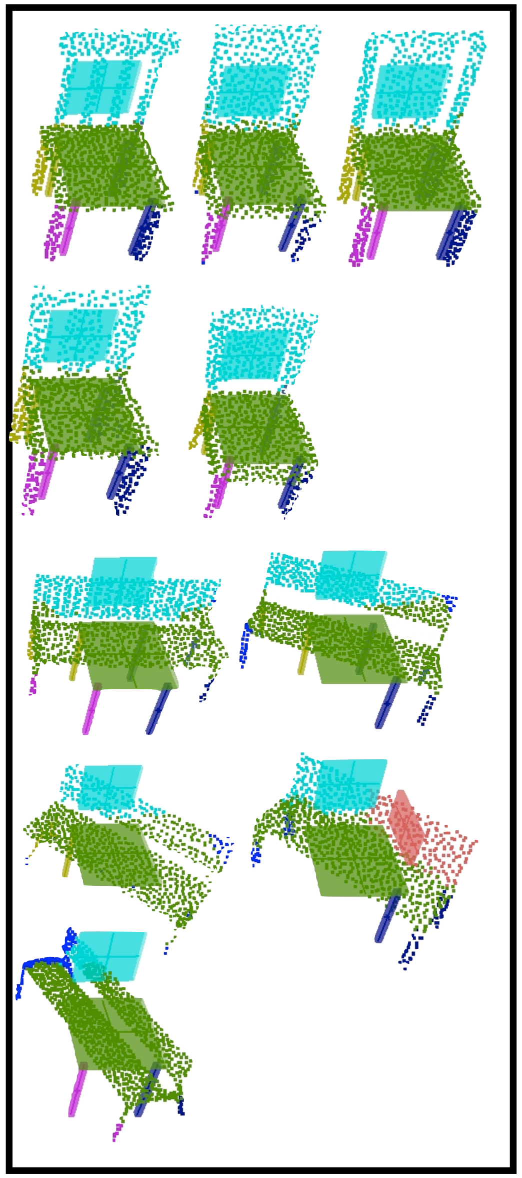
a. Initial Template

Overview

Improve set of templates from unlabeled geometry



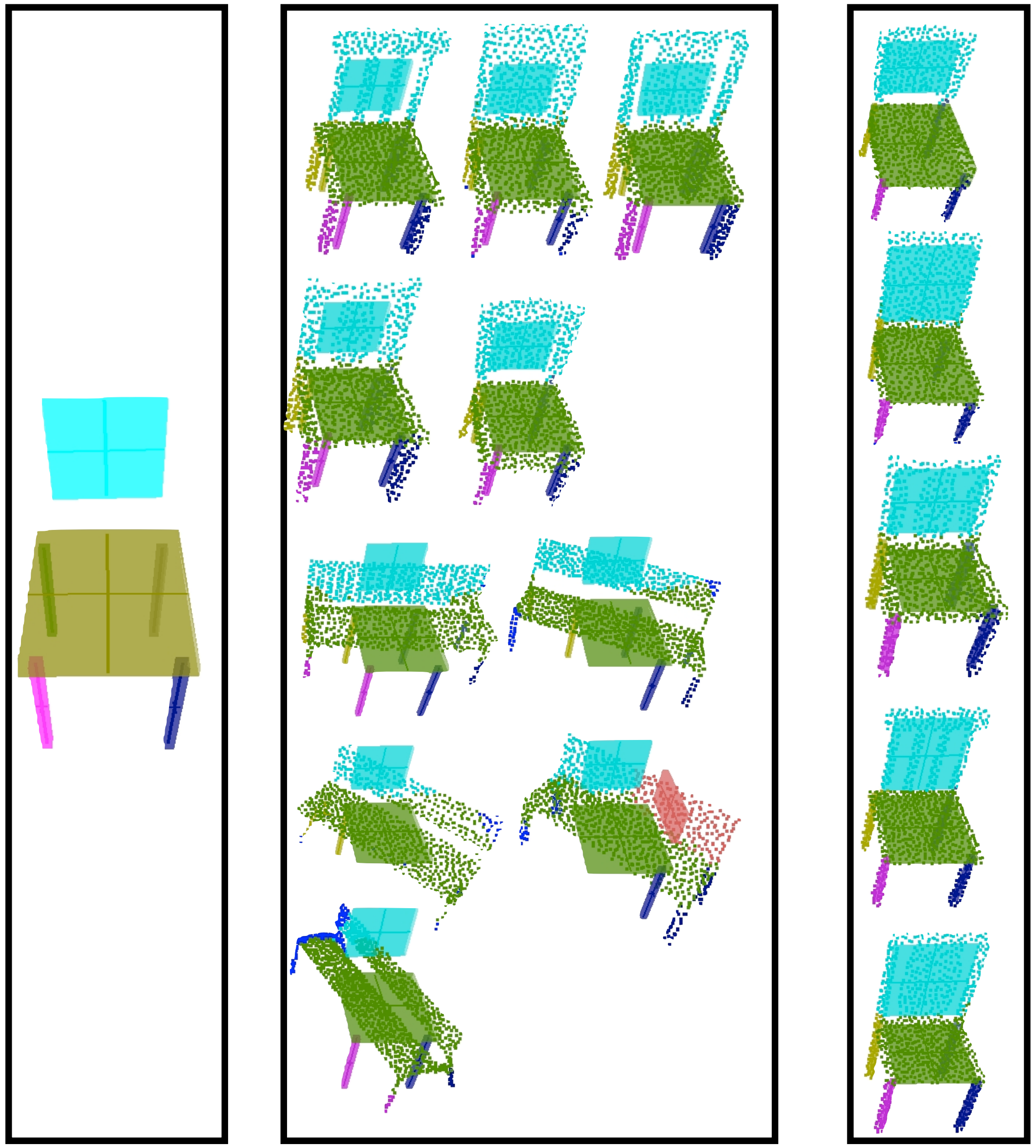
a. Initial Template



b. Fitting Set

Overview

Improve set of templates from unlabeled geometry



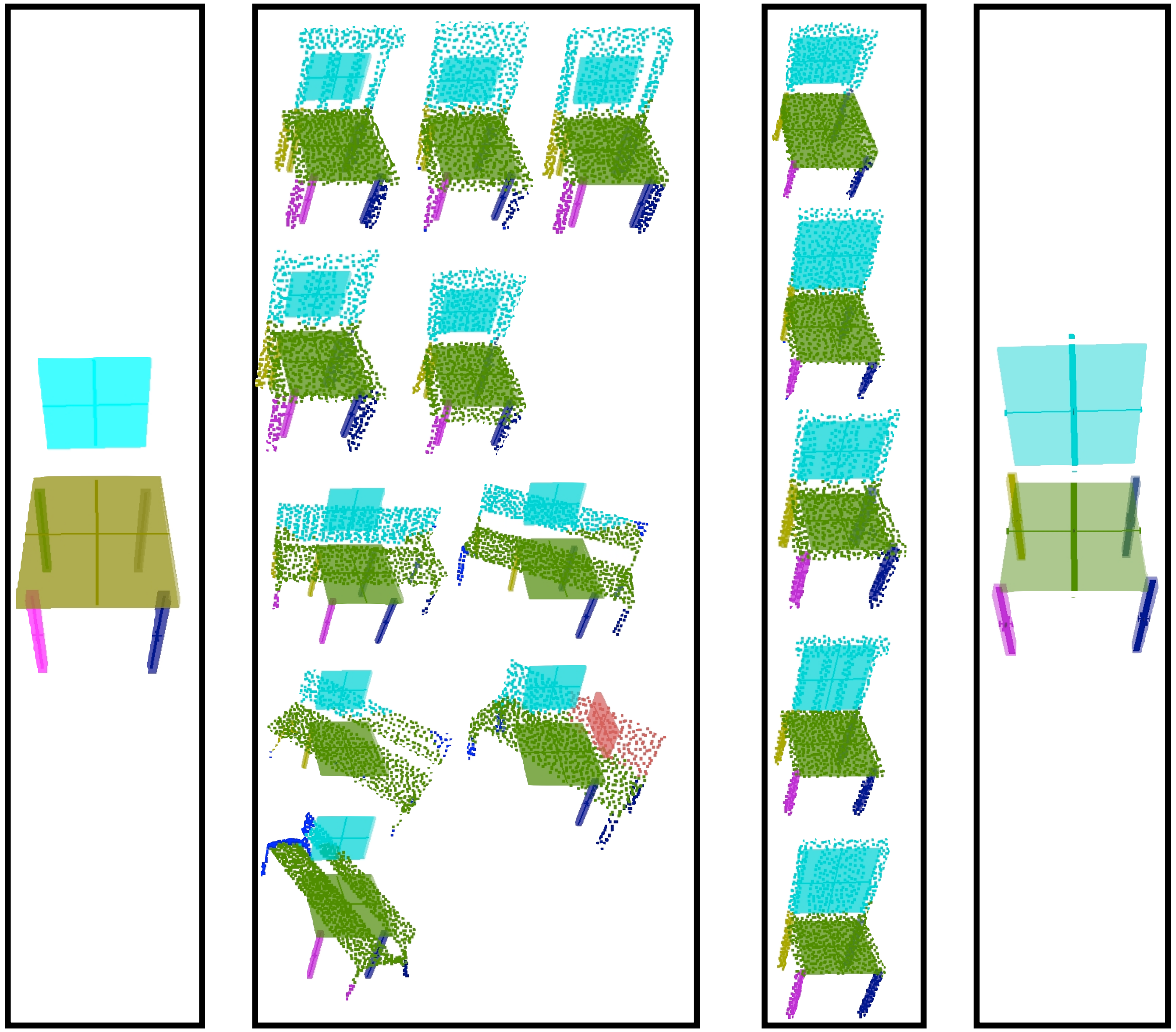
a. Initial Template

b. Fitting Set

c. Learning Set

Overview

Improve set of templates from unlabeled geometry



a. Initial Template

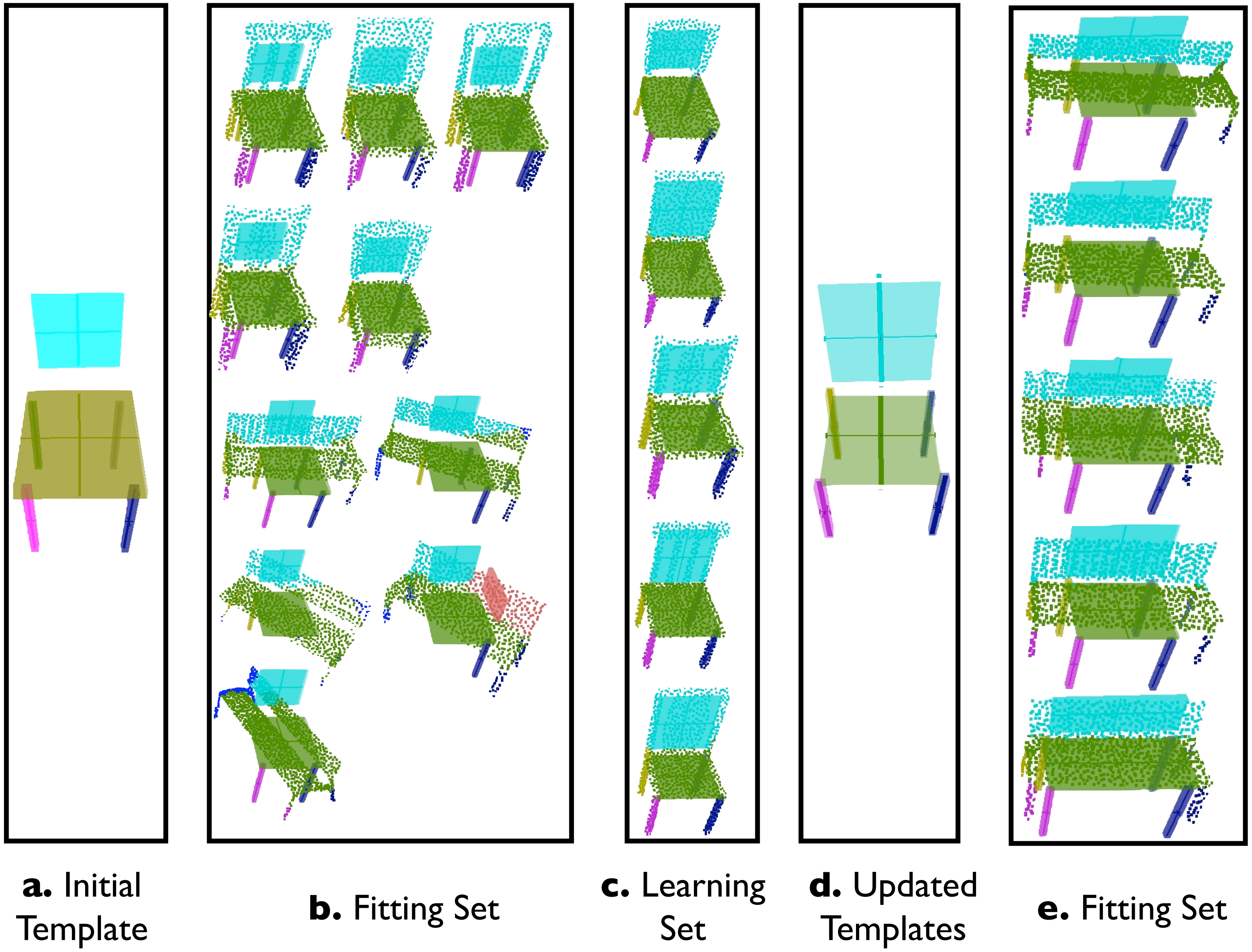
b. Fitting Set

c. Learning Set

d. Updated Templates

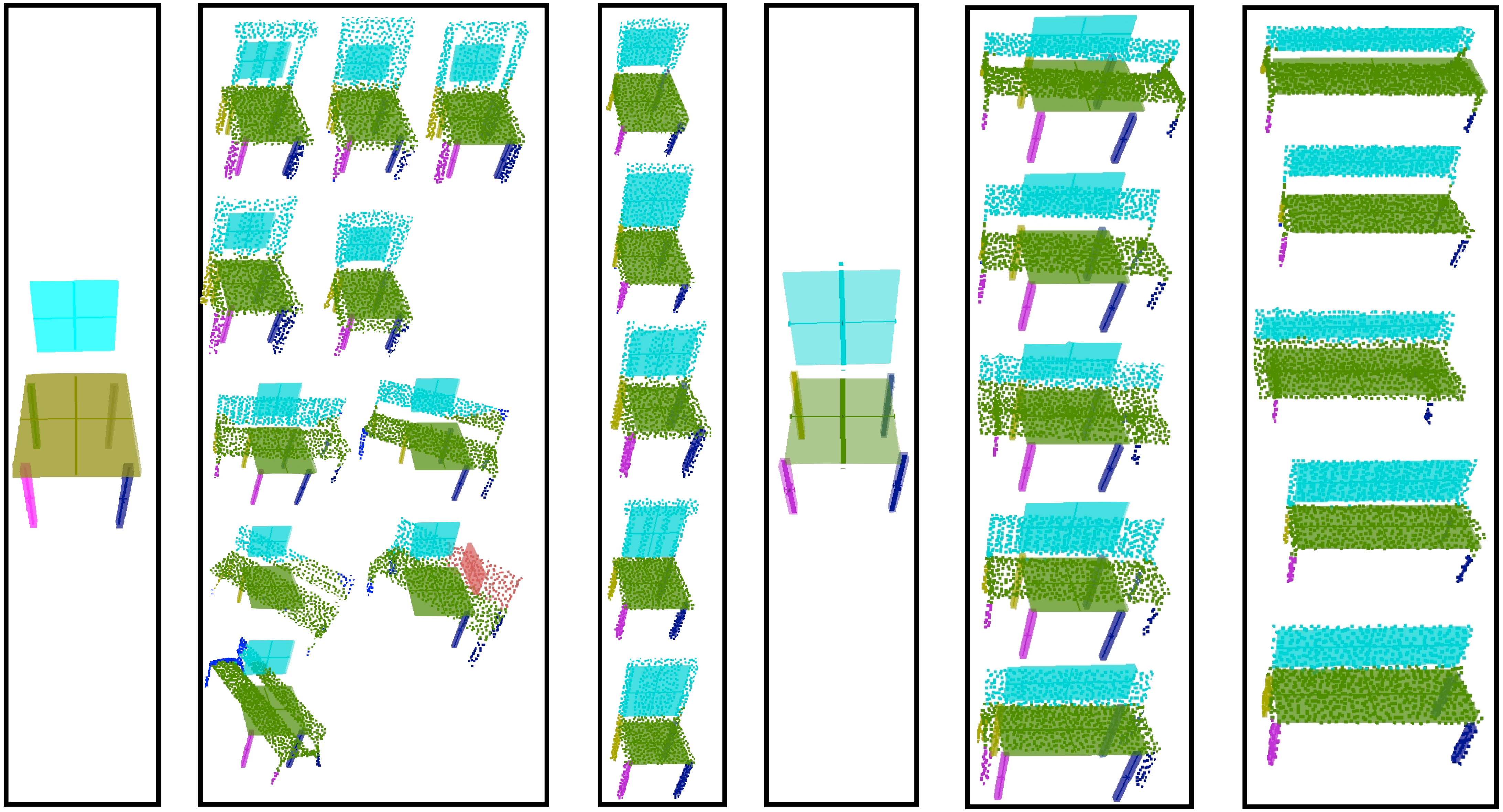
Overview

Improve set of templates from unlabeled geometry



Overview

Improve set of templates from unlabeled geometry



a. Initial Template

b. Fitting Set

c. Learning Set

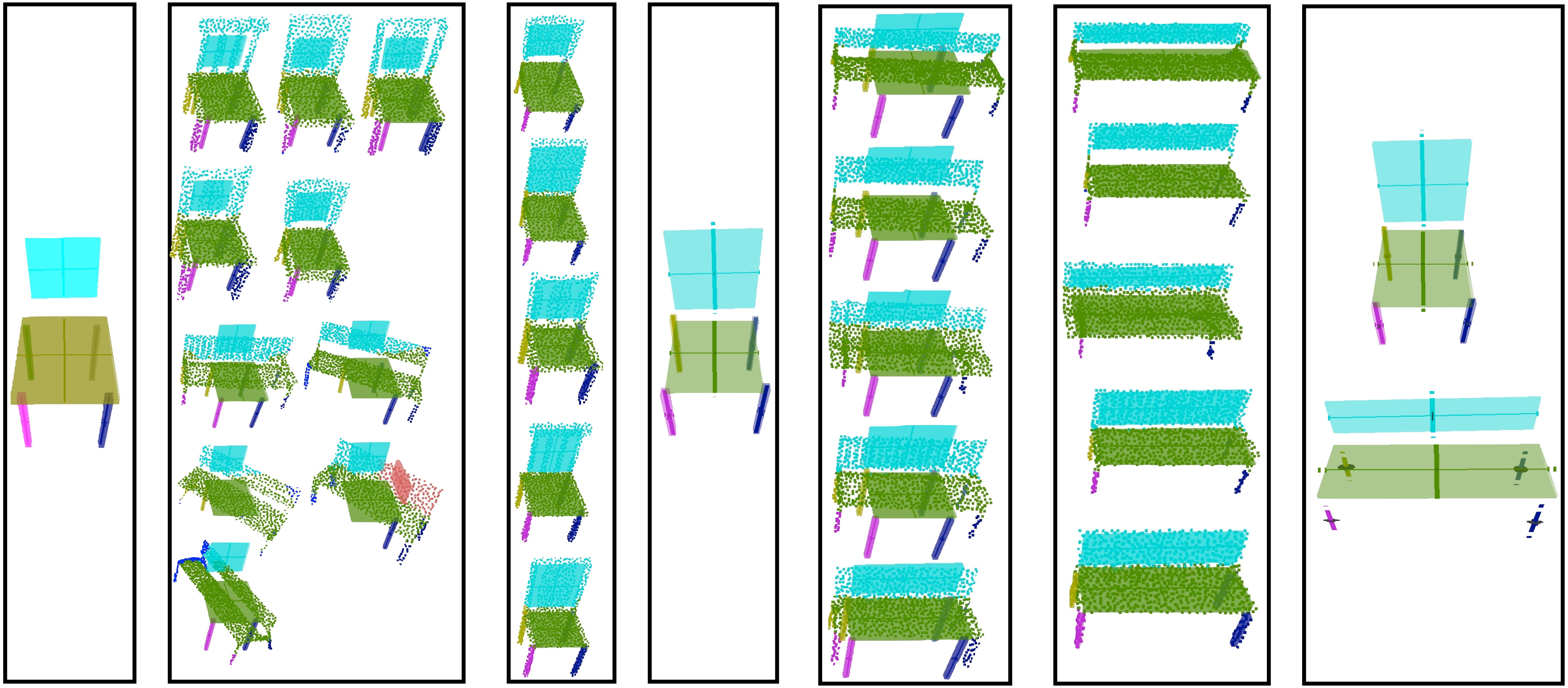
d. Updated Templates

e. Fitting Set

f. Learning Set

Overview

Improve set of templates from unlabeled geometry



a. Initial Template

b. Fitting Set

c. Learning Set

d. Updated Templates

e. Fitting Set

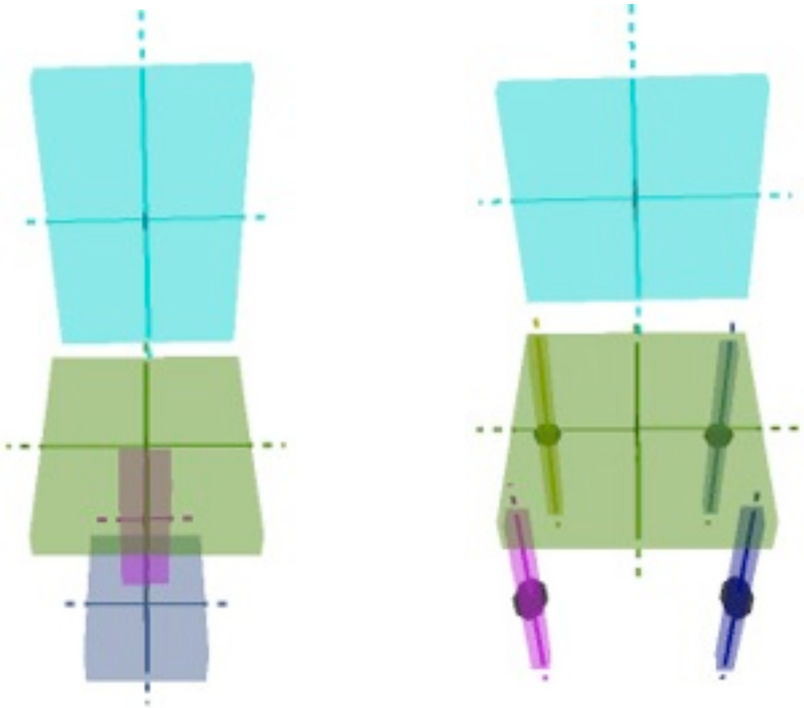
f. Learning Set

g. Updated Templates

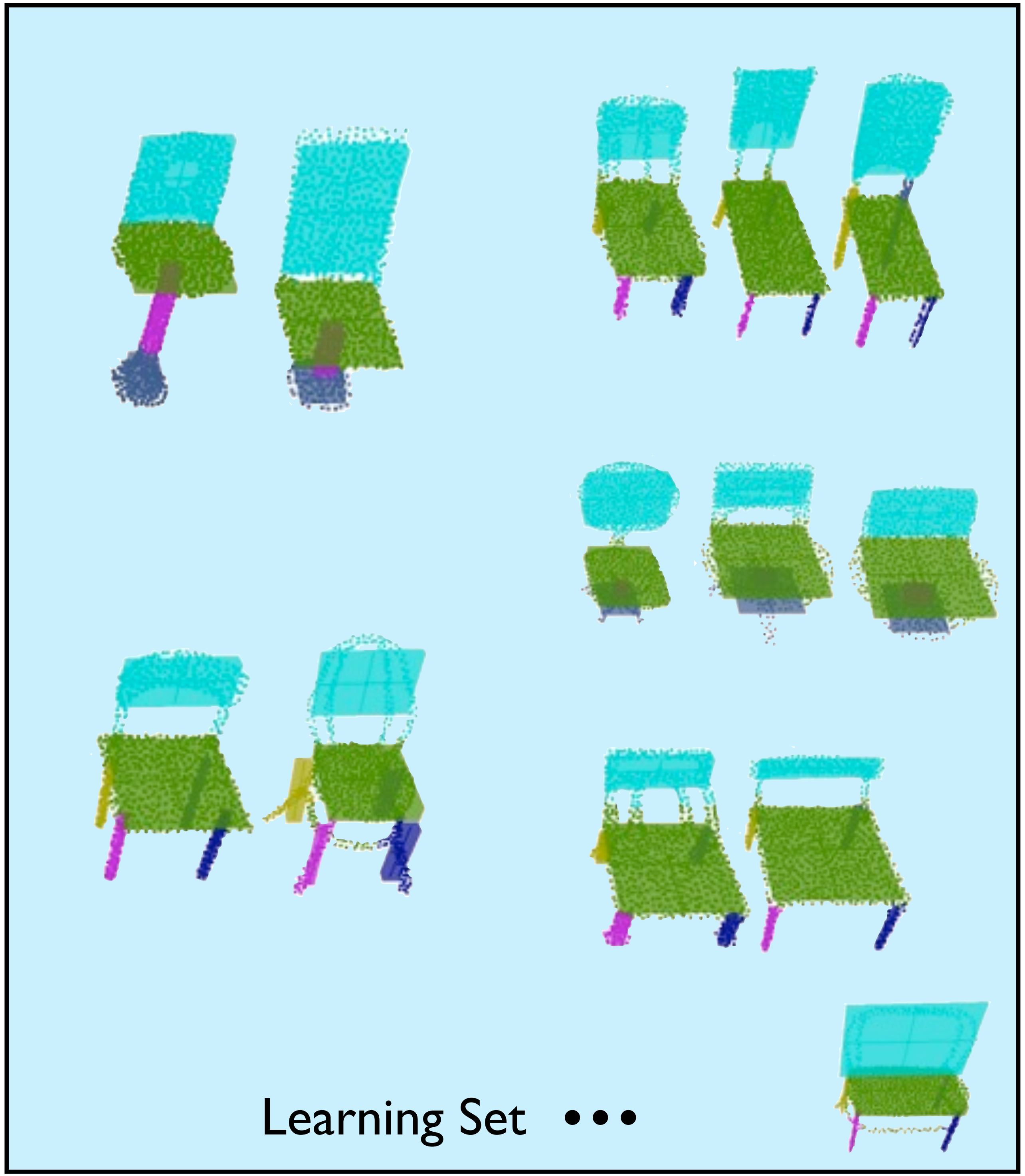
Template Refinement

Update template set from deformations in Learning Set

- Update current
- Spawn new
- Reject outliers



Current Template Set

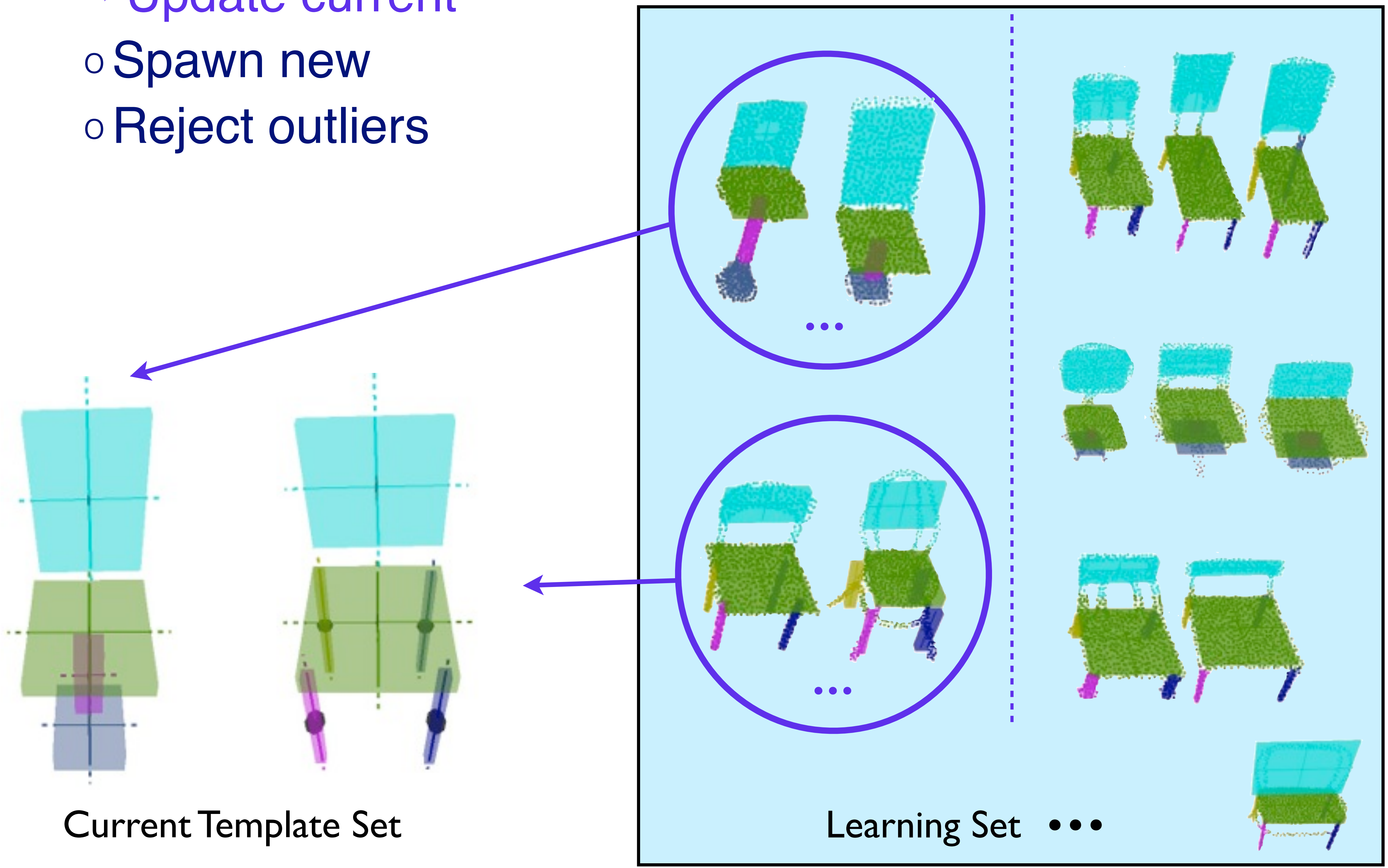


Learning Set •••

Template Refinement

Update template set from deformations in Learning Set

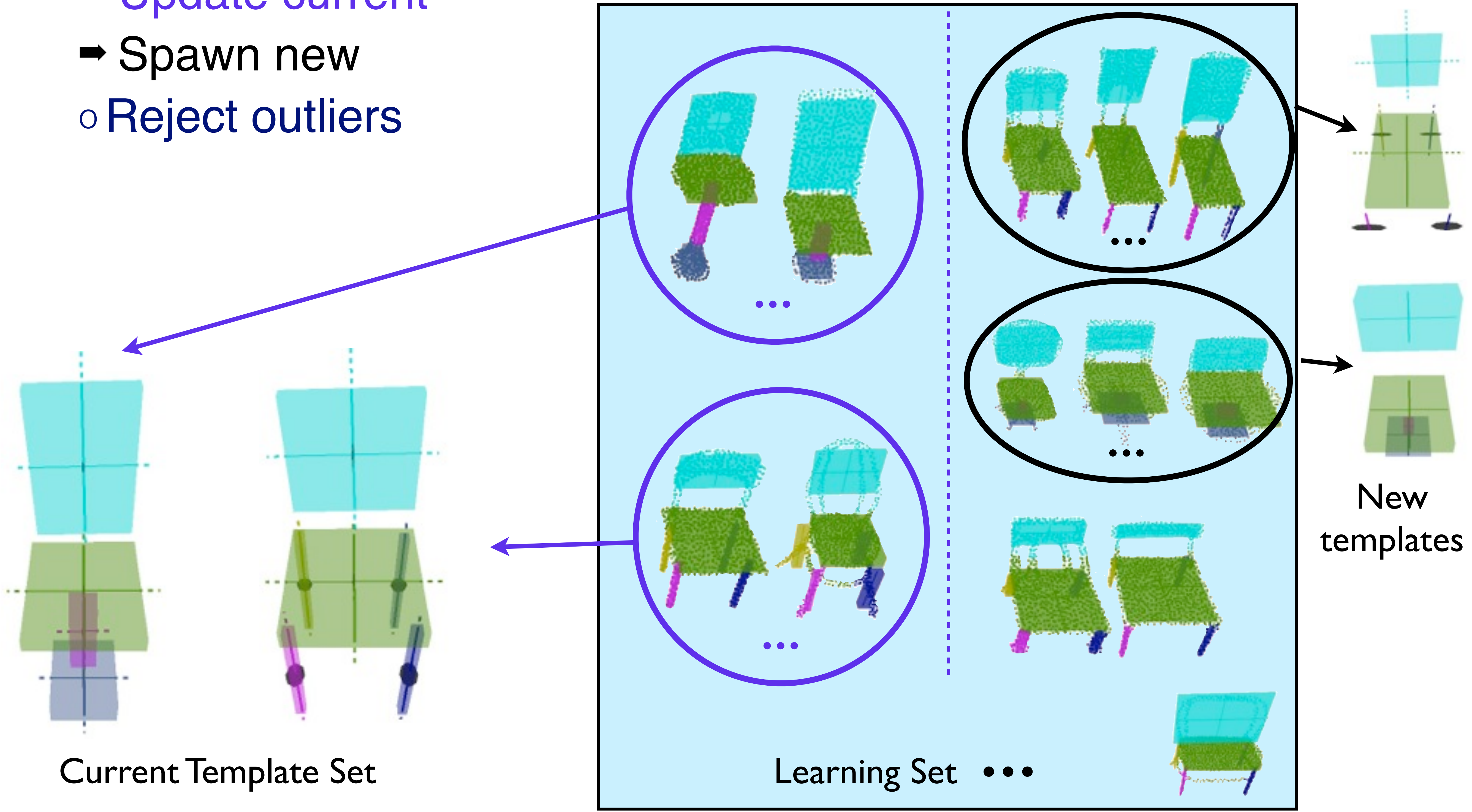
- ➡ Update current
- o Spawn new
- o Reject outliers



Template Refinement

Update template set from deformations in Learning Set

- ➡ Update current
- ➡ Spawn new
- o Reject outliers



Current Template Set

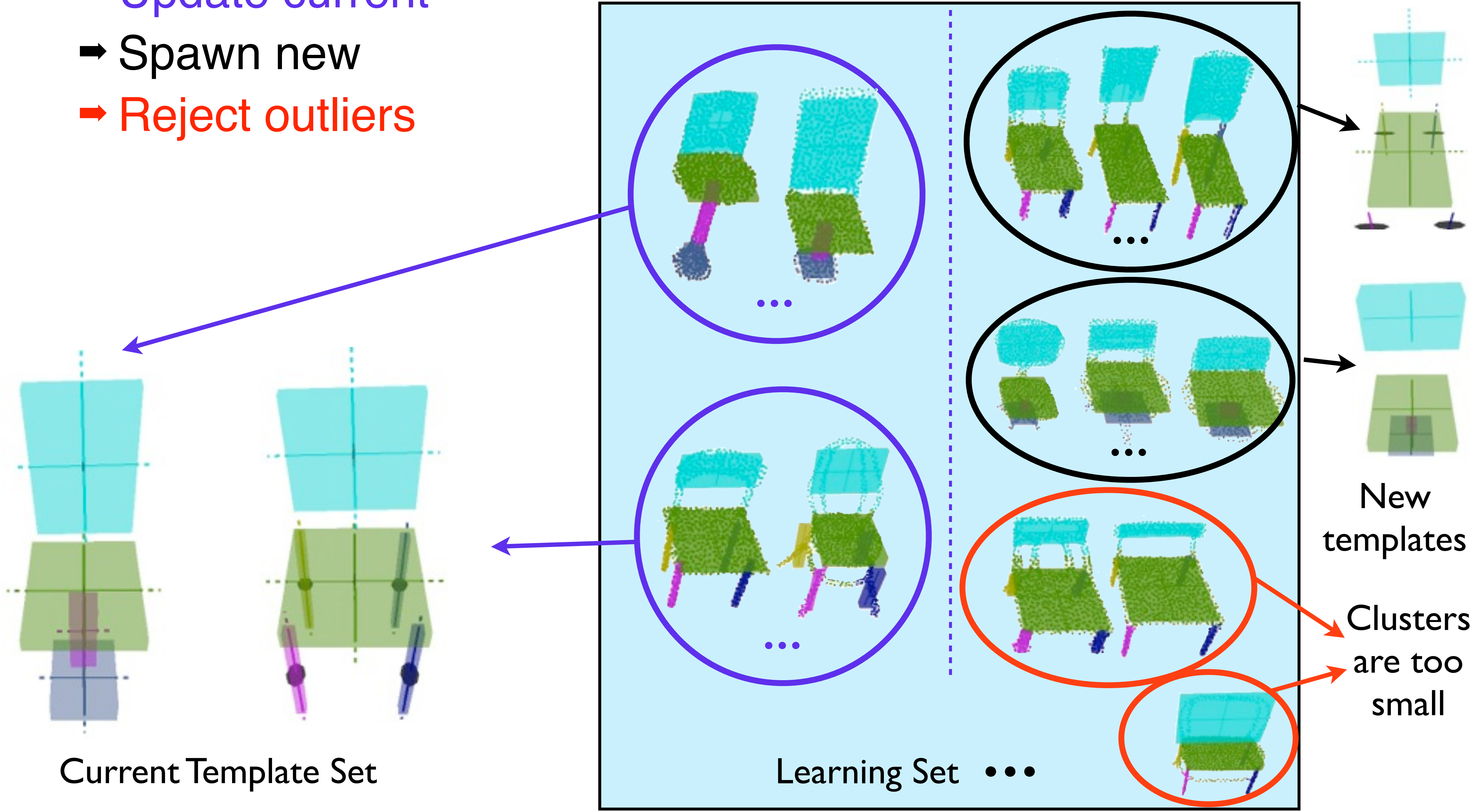
Learning Set •••

New templates

Template Refinement

Update template set from deformations in Learning Set

- ➡ Update current
- ➡ Spawn new
- ➡ Reject outliers



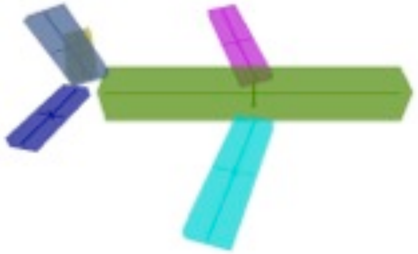
Results

Evaluation

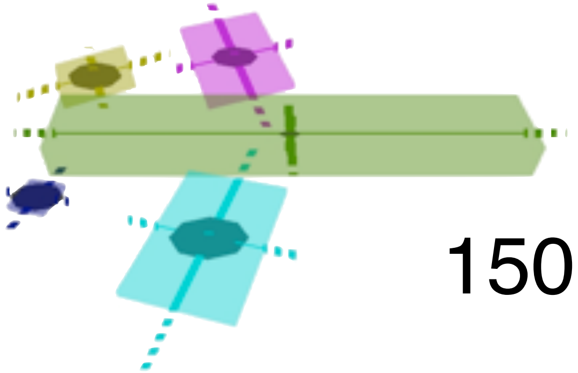
- ➡ Examples
- o Correspondence benchmark
- o Segmentation benchmark
- o Timing and complexity

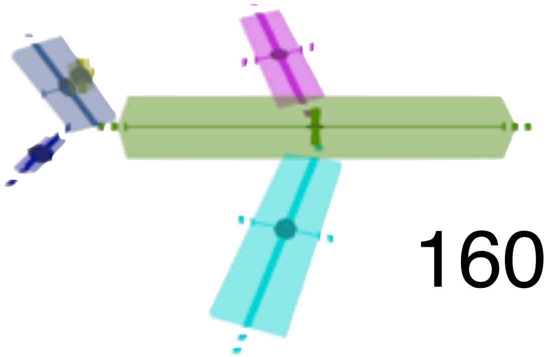
3D Warehouse planes

Initial Template:

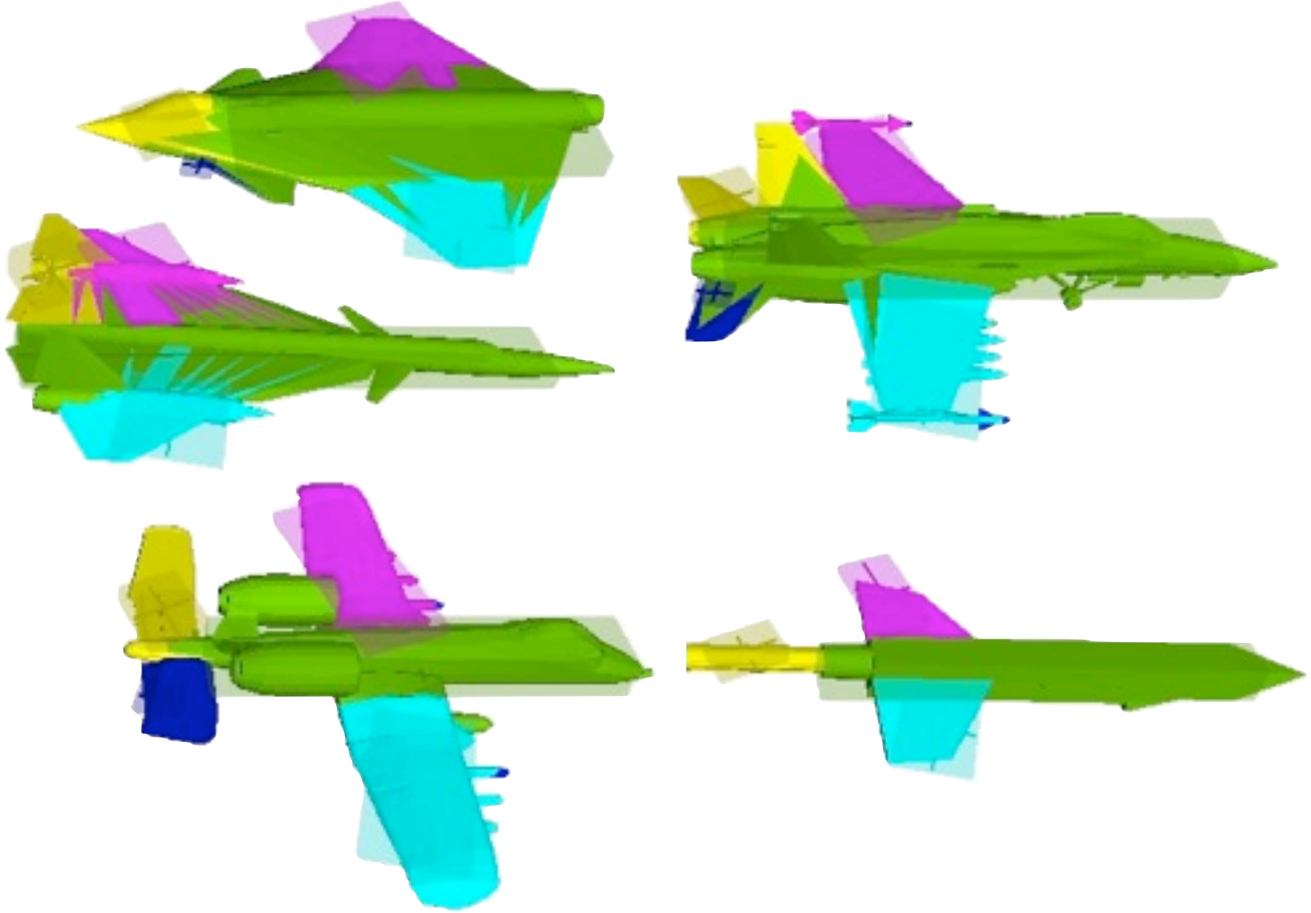


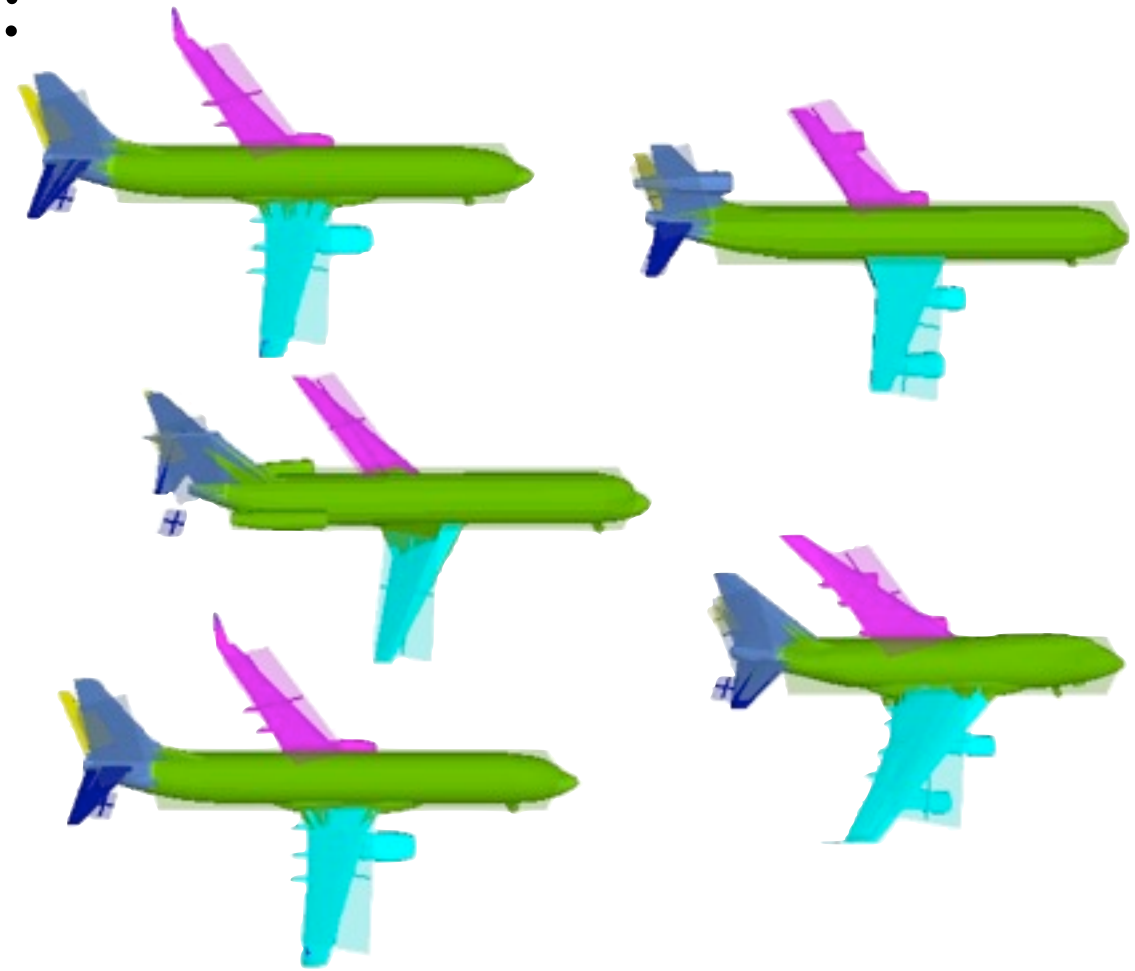
Final Templates:





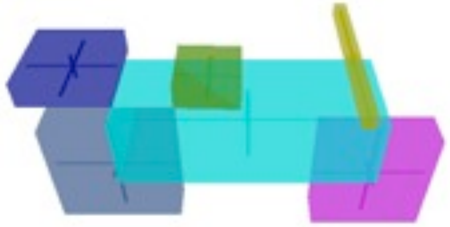
Randomly sampled template fitting results:

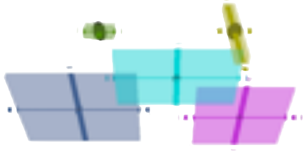




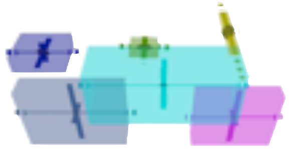
3D Warehouse bikes

Initial Template:



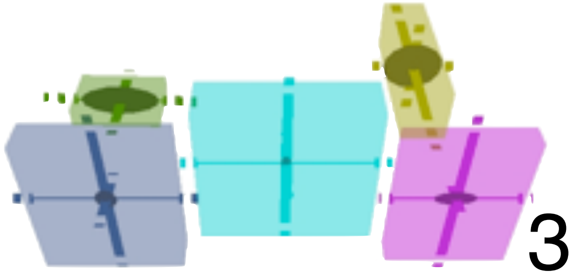


23

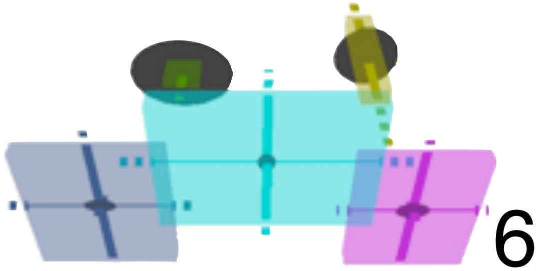


7

Final Templates:



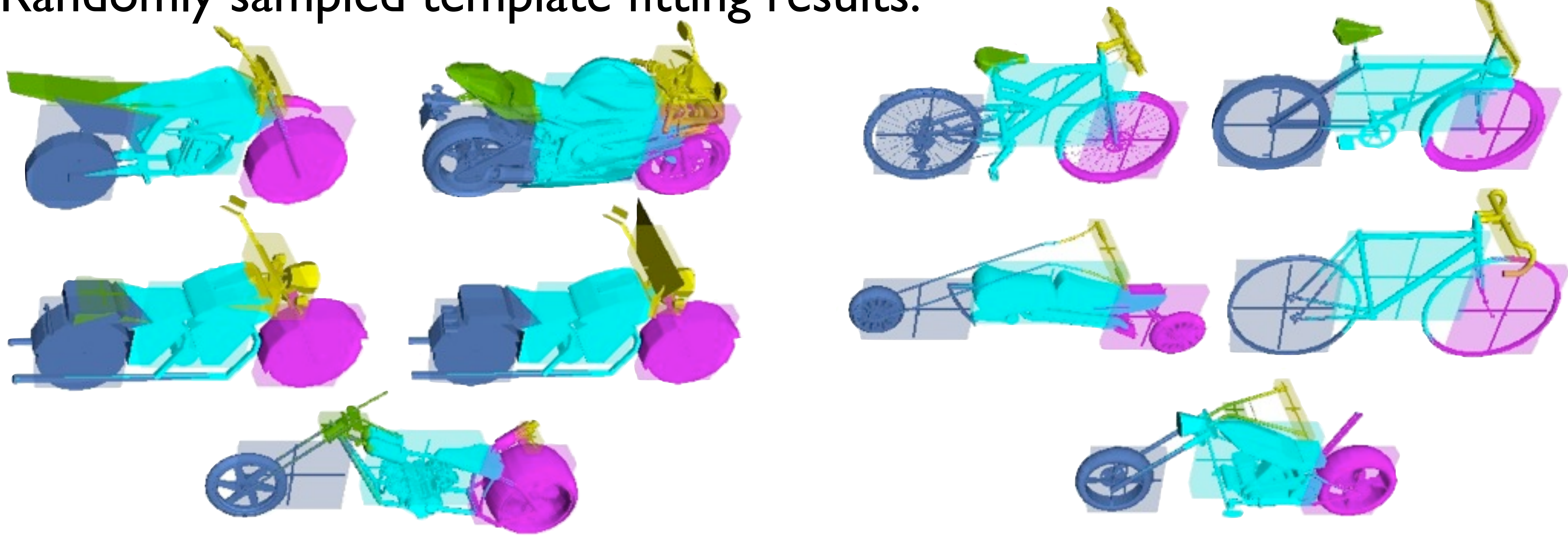
378



63

•••

Randomly sampled template fitting results:

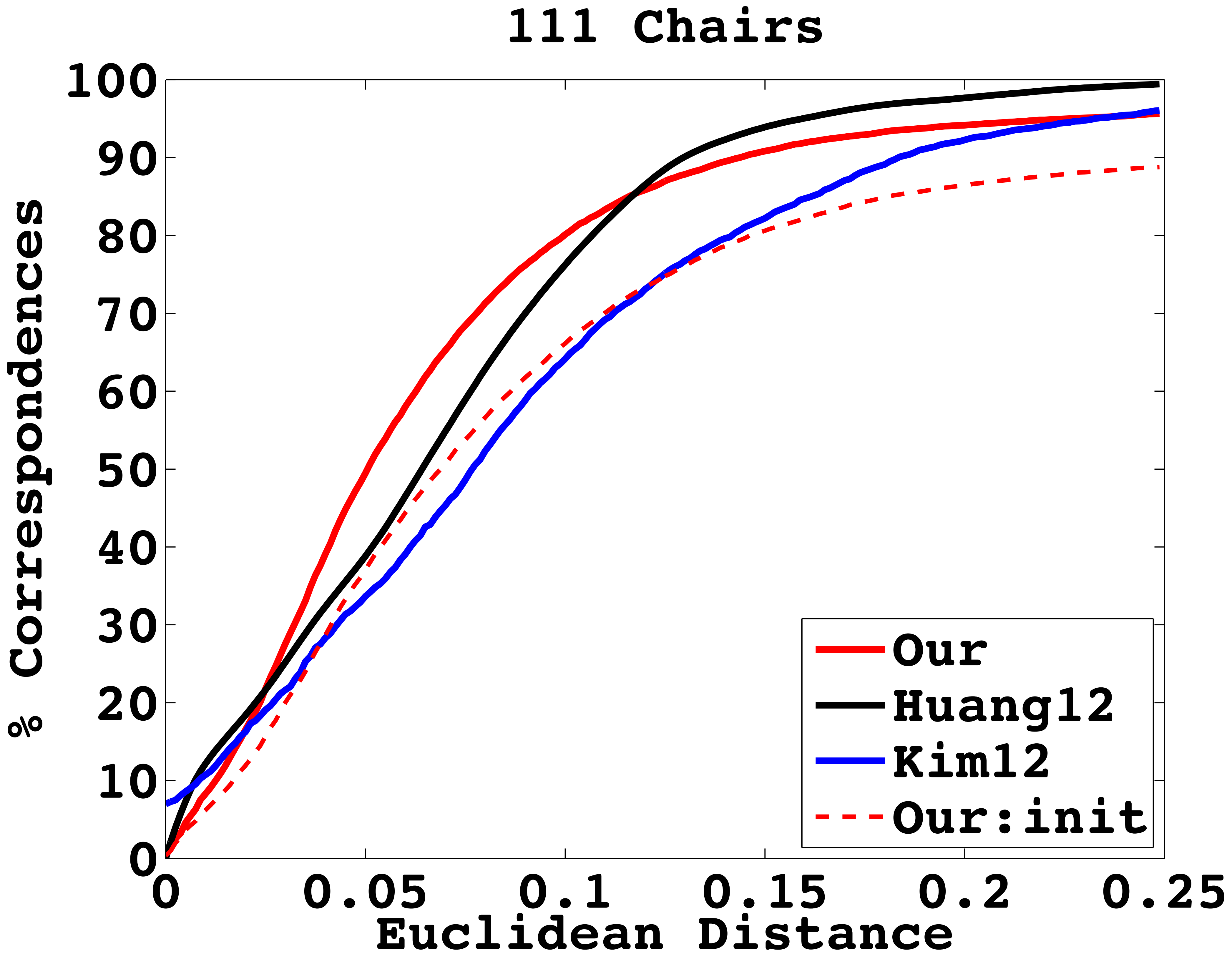


Results

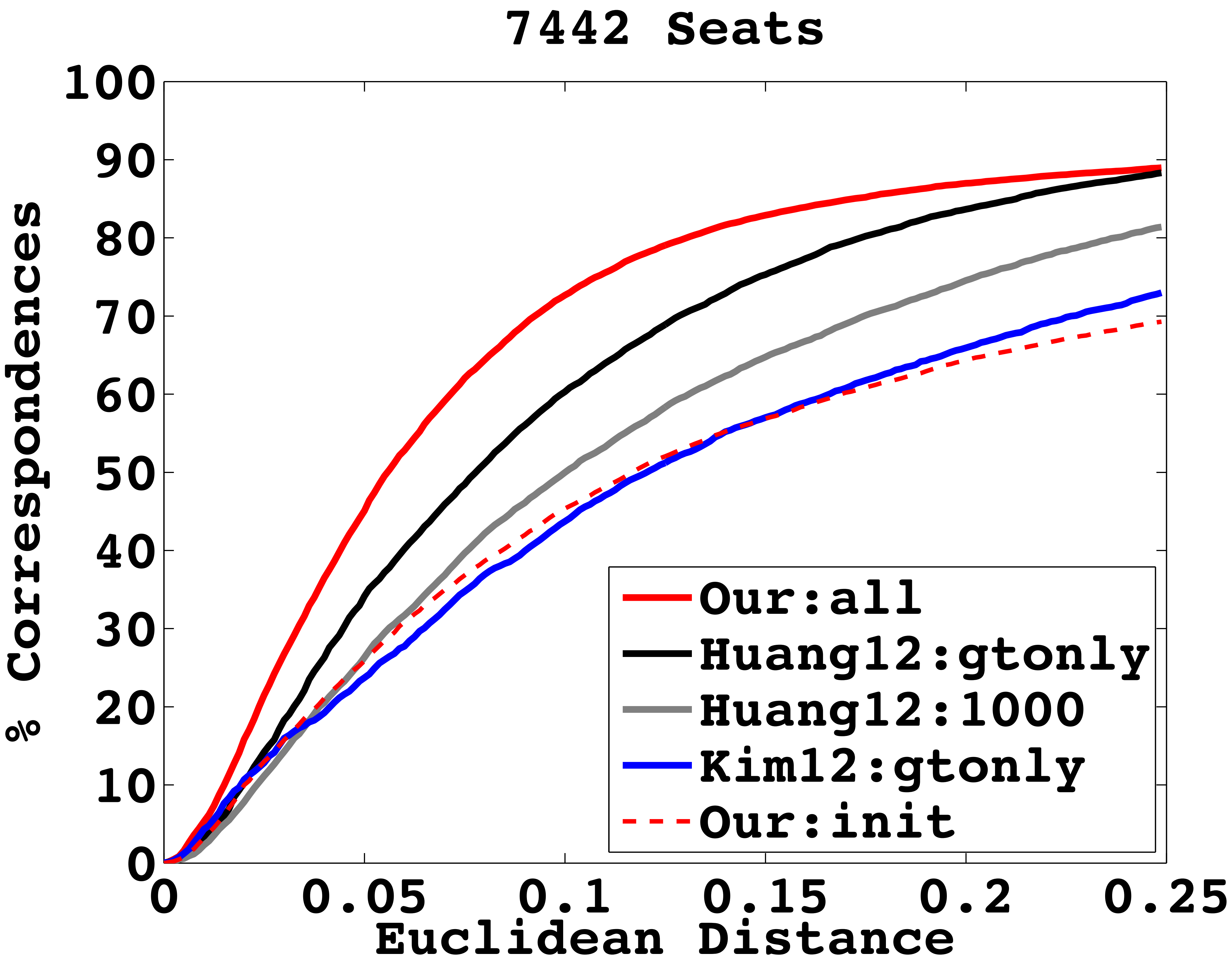
Evaluation

- o Examples
- ➡ Correspondence benchmark
- o Segmentation benchmark
- o Timing and complexity

Correspondences benchmark



Correspondences benchmark



Results

Evaluation

- o Examples
- o Correspondence benchmark
- ➡ Segmentation benchmark
- o Timing and complexity

Co-segmentation benchmark

Class	Hu	Auto result
Chairs	89.6	97.6
Lamps	90.7	95.2
FourLegged	88.7	86.9
Goblets	99.2	97.6
Vase	80.2	81.3
Guitars	98.0	88.5
Candelabra	93.9	82.4

Same or ours is better

Results

Evaluation

- o Examples
- o Correspondence benchmark
- o Segmentation benchmark
- ➡ Timing and complexity

Timing and complexity

Timing

- 20 shapes: 2-3min
- 100 shapes: 10-30 min
- 3000 planes: 3.3 hrs
- 7000 chairs: 10 hrs

Timing and complexity

Timing

- 20 shapes: 2-3min
- 100 shapes: 10-30 min
- 3000 planes: 3.3 hrs
- 7000 chairs: 10 hrs

Complexity

N - collection size,
K_L - learning set size,
T_{max} - number of templates

- At most O(N) iterations
- Each iteration is $O(K_L \; T_{max} + K_L^2)$

Summary

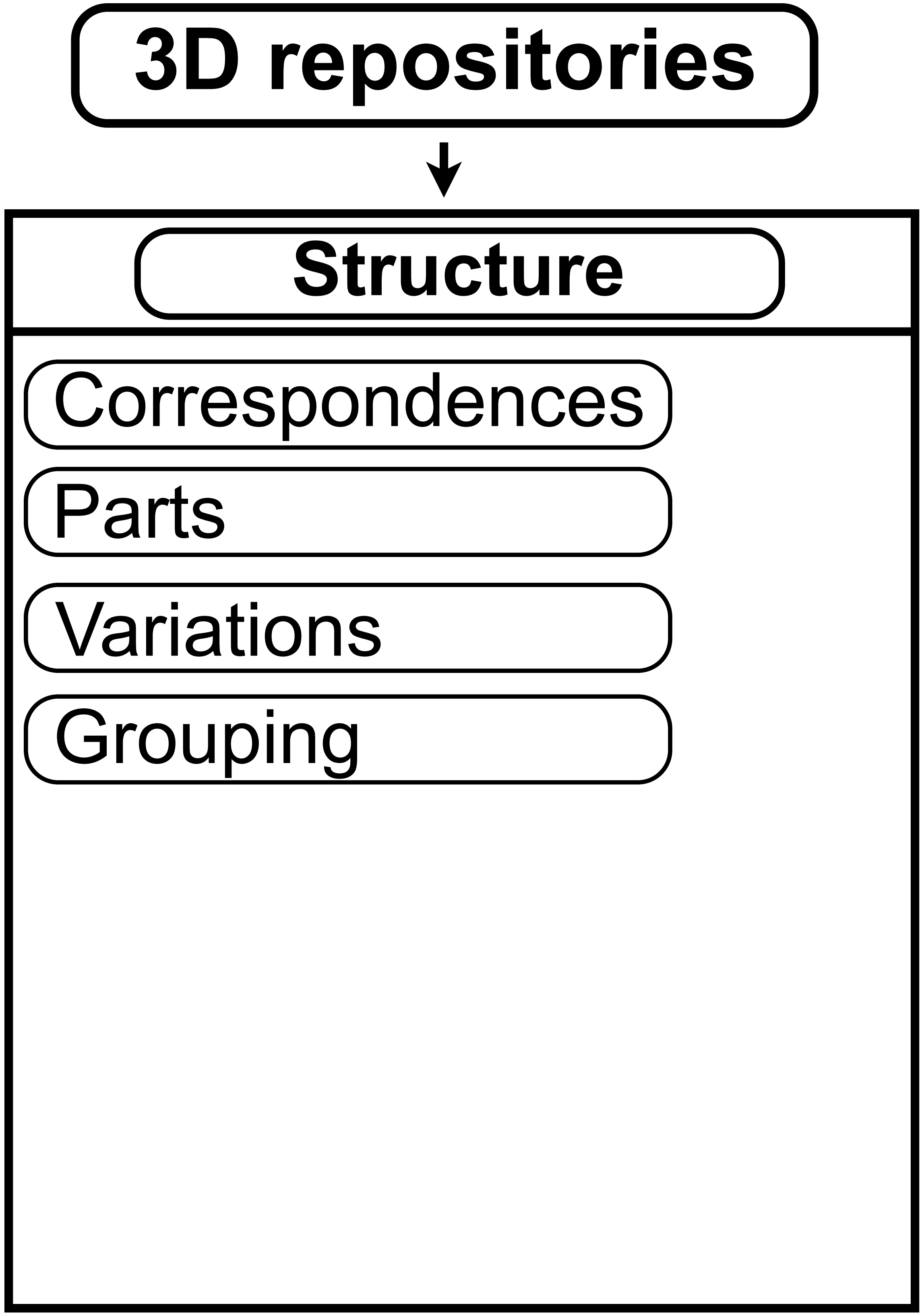
Given a collection, we jointly:

- Cluster models
- Learn a part-based deformable model
- Compute consistent segmentations
- Compute correspondences

Our algorithm is:

- Linear in size of collection
- Out-of-core
- Performs favorably on benchmark datasets

Recap + References



3D repositories

Structure

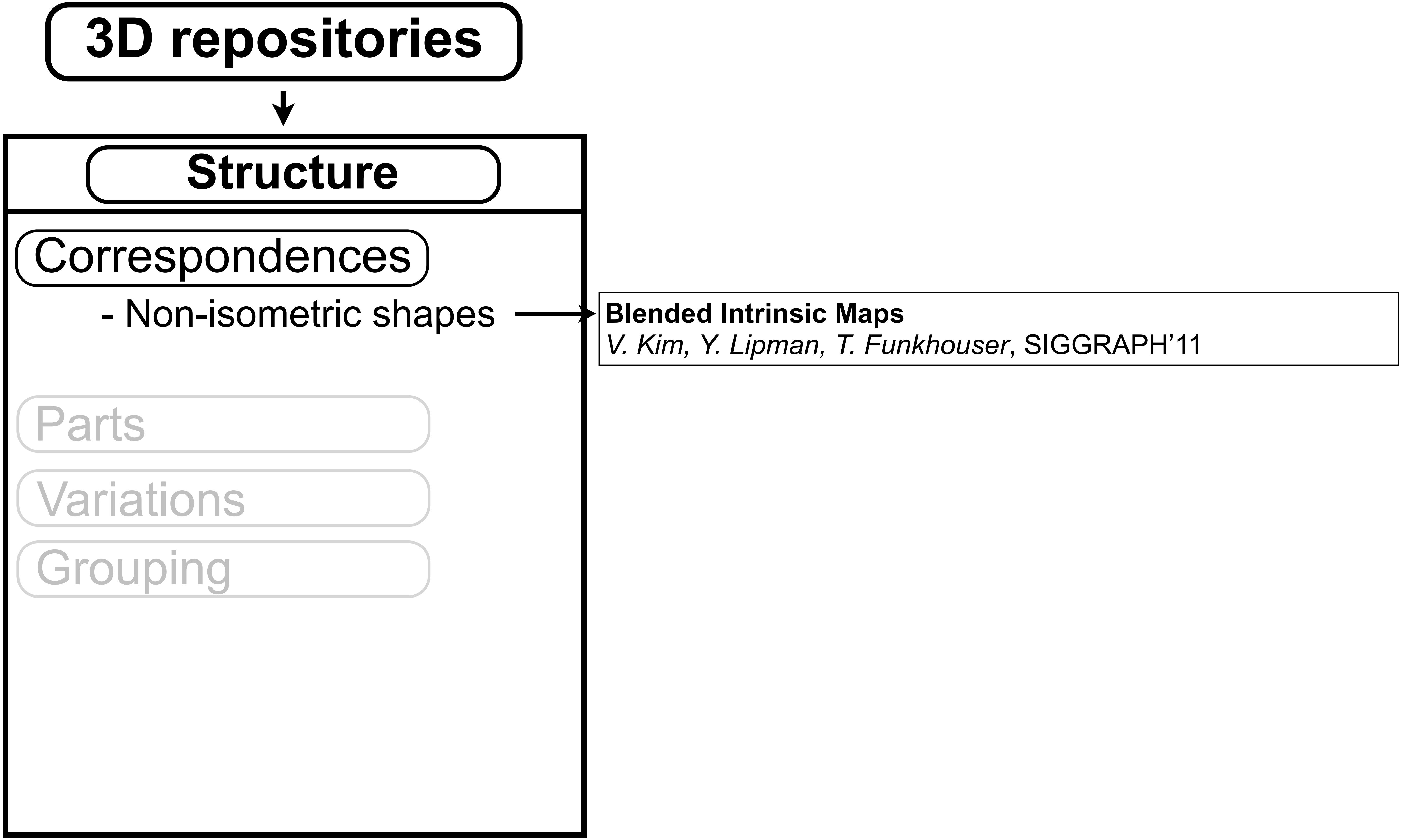
Correspondences

Parts

Variations

Grouping

Recap + References



3D repositories

Structure

Correspondences

- Non-isometric shapes

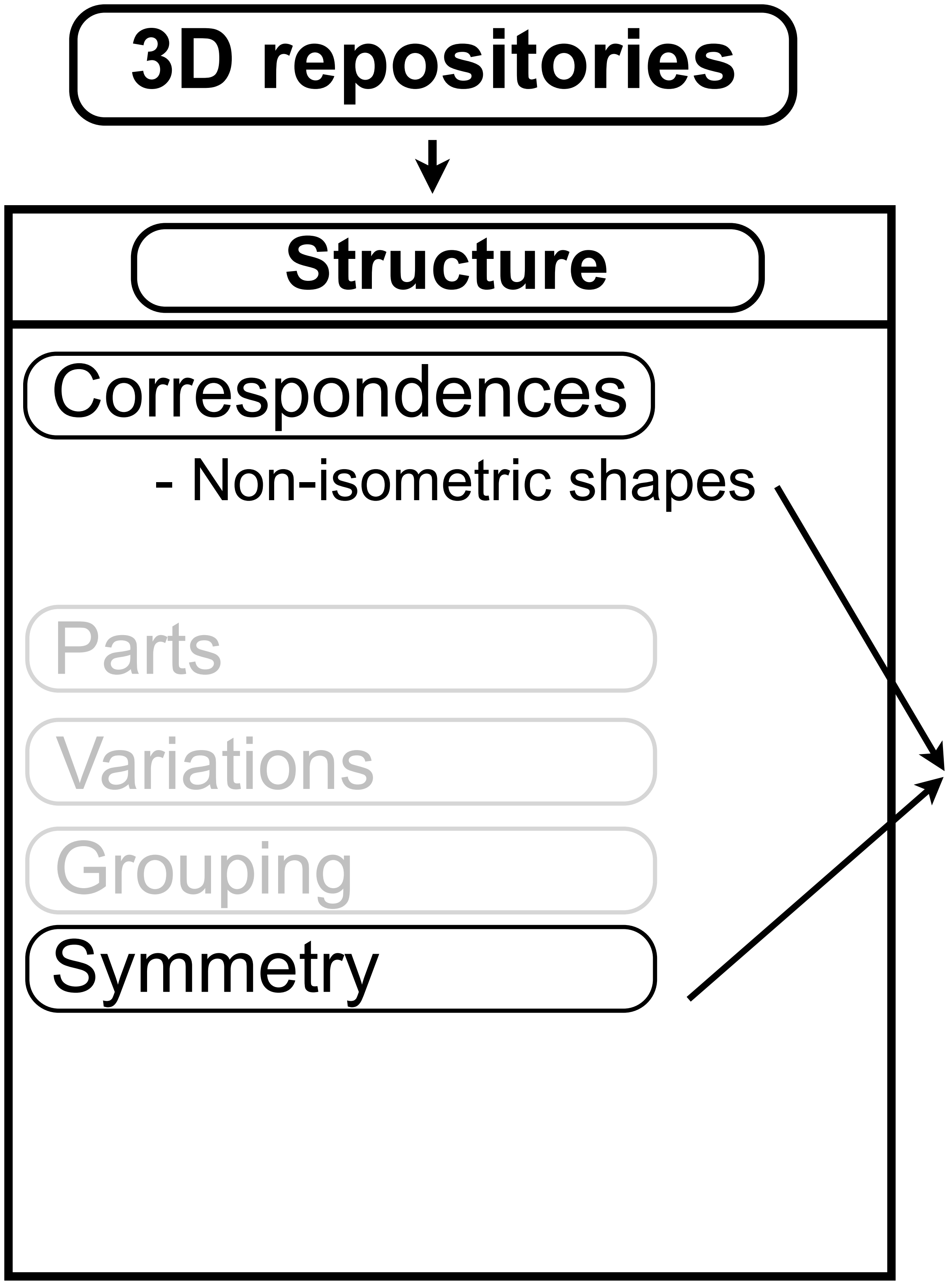
Blended Intrinsic Maps
V. Kim, Y. Lipman, T. Funkhouser, SIGGRAPH’11

Parts

Variations

Grouping

Recap + References



3D repositories

Structure

Correspondences

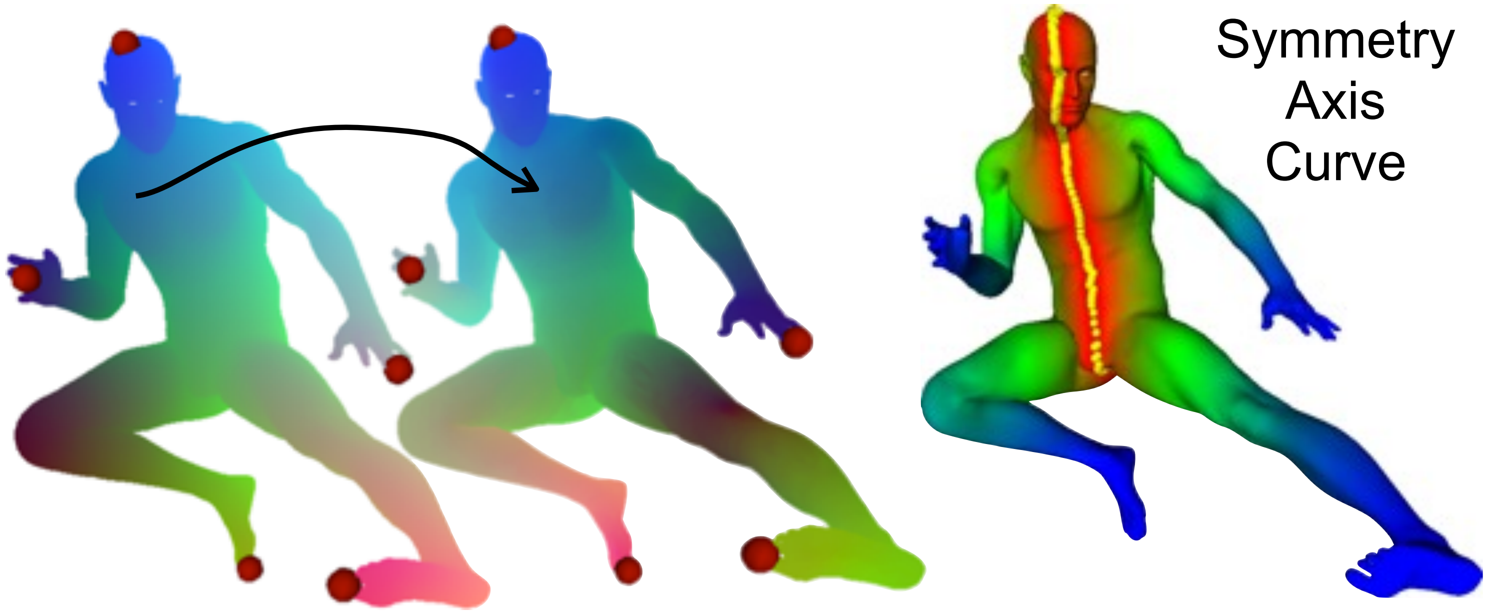
- Non-isometric shapes

Parts

Variations

Grouping

Symmetry



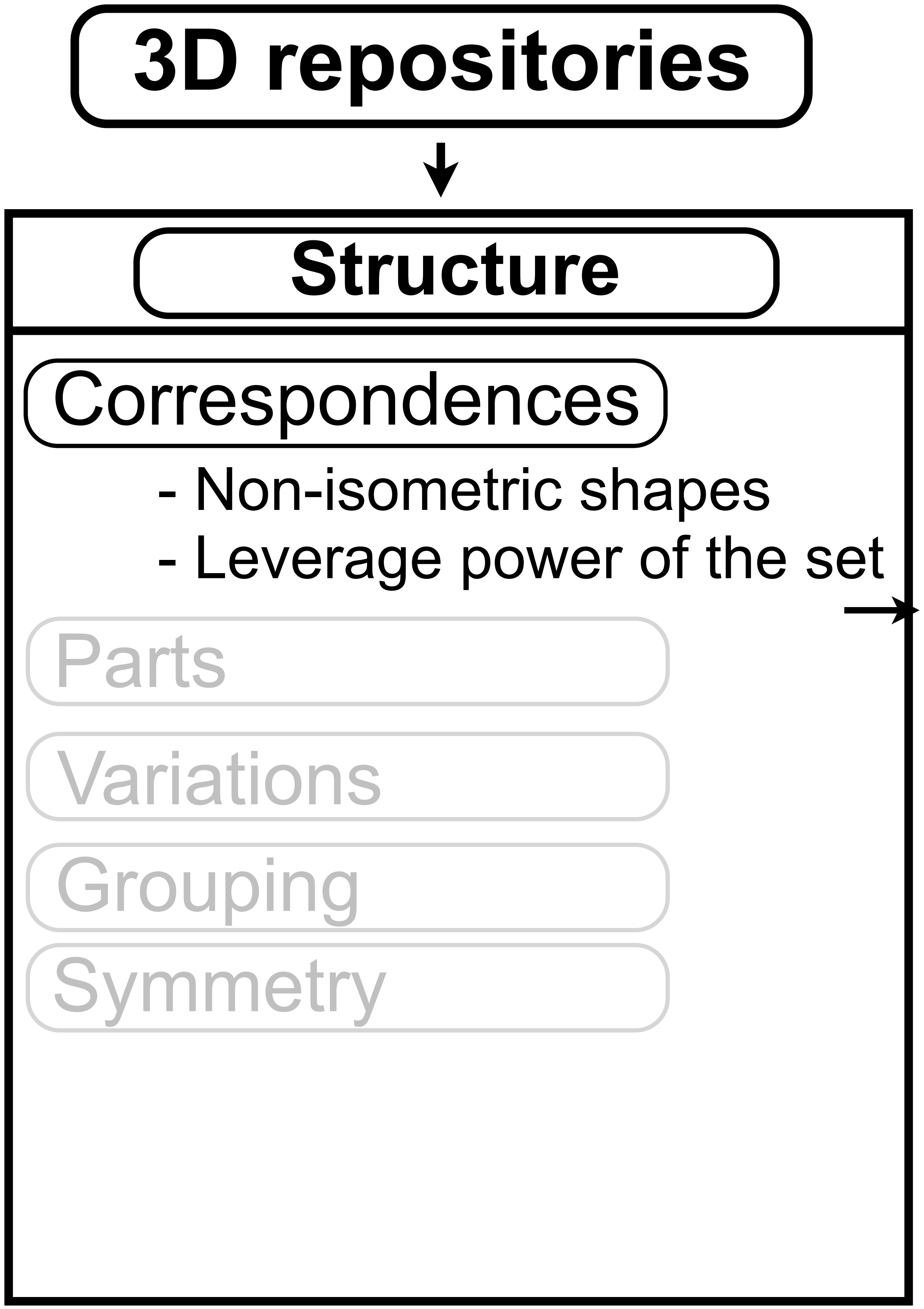
(c) Liu et al., SGP'12

Can use our correspondence-detection technique:

Möbius Transformations
for Global Intrinsic Symmetry Analysis
V. Kim, Y. Lipman, X. Chen, T. Funkhouser, SGP'10

Finding Surface Correspondences
using Symmetry Axis Curves
T. Liu, V. Kim, T. Funkhouser, SGP'12

Recap + References



3D repositories

Structure

Correspondences

- Non-isometric shapes
- Leverage power of the set

Parts

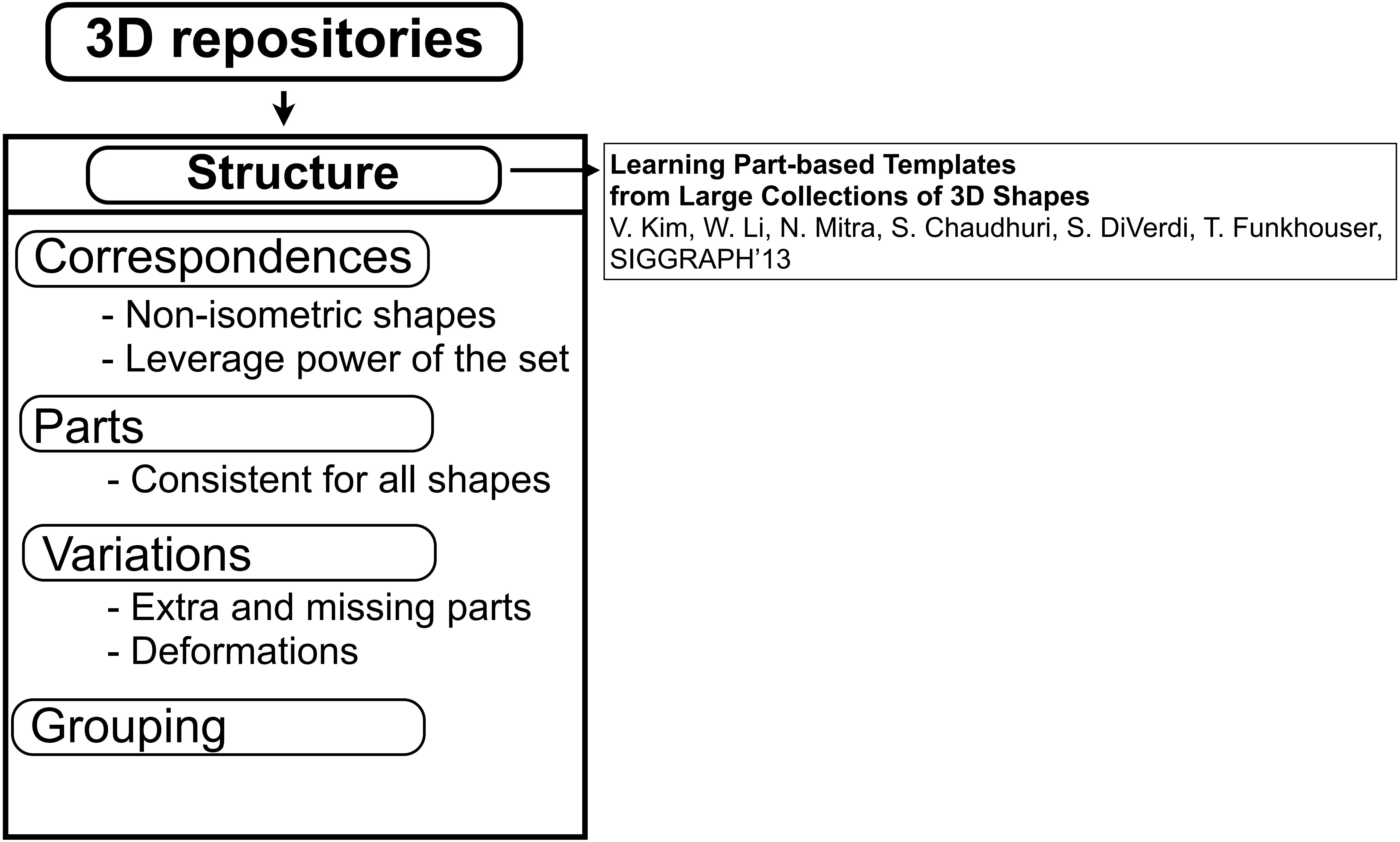
Variations

Grouping

Symmetry

Exploring Collections of 3D Models using Fuzzy Correspondences
V. Kim, W. Li, N. Mitra, S. DiVerdi, T. Funkhouser, SIGGRAPH’12

Recap + References



3D repositories

Structure

Correspondences

- Non-isometric shapes
- Leverage power of the set

Parts

- Consistent for all shapes

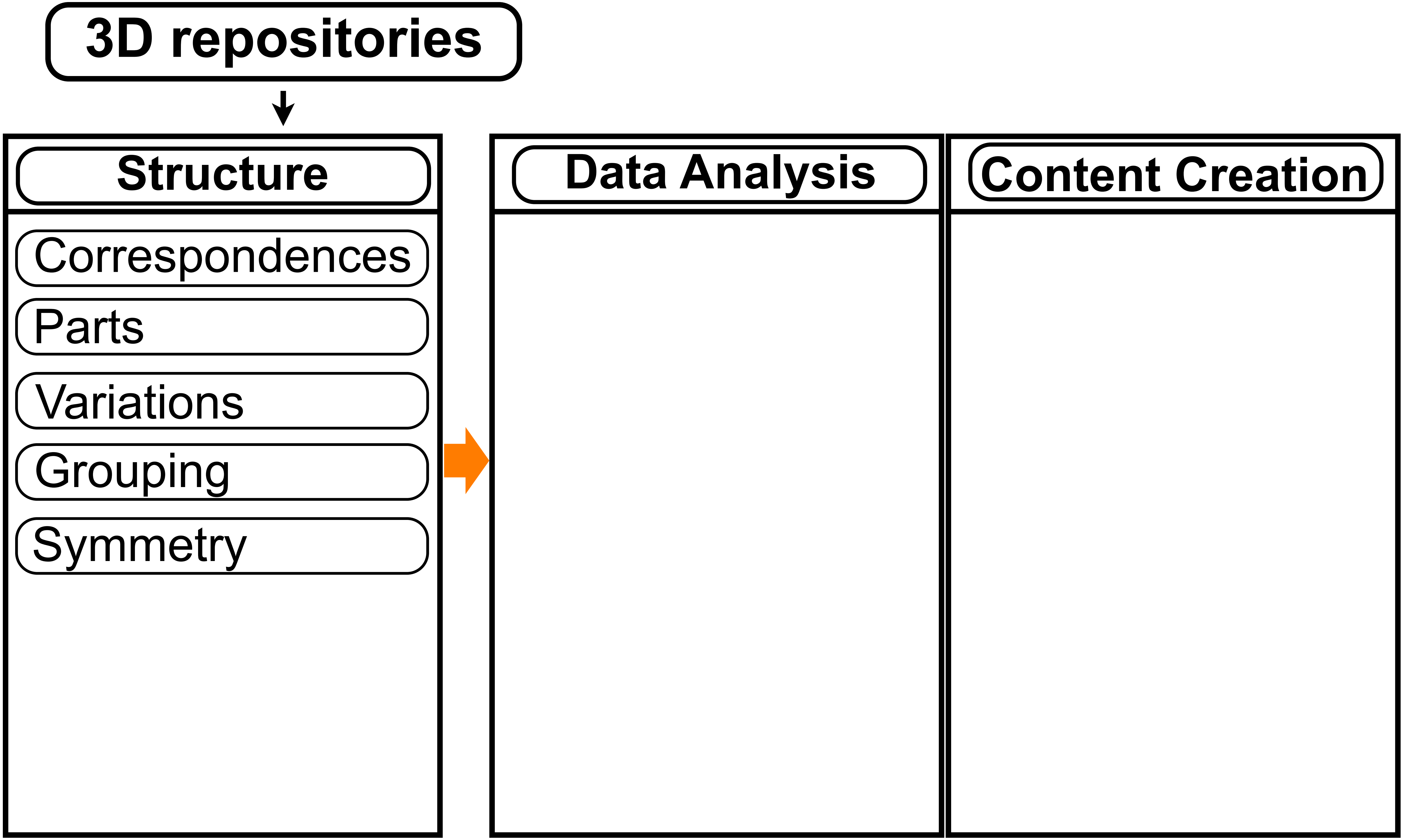
Variations

- Extra and missing parts
- Deformations

Grouping

Learning Part-based Templates from Large Collections of 3D Shapes
V. Kim, W. Li, N. Mitra, S. Chaudhuri, S. DiVerdi, T. Funkhouser, SIGGRAPH'13

Recap + References



3D repositories

Structure

Correspondences

Parts

Variations

Grouping

Symmetry

Data Analysis

Content Creation

Recap + References

3D repositories

Structure

Correspondences

Parts

Variations

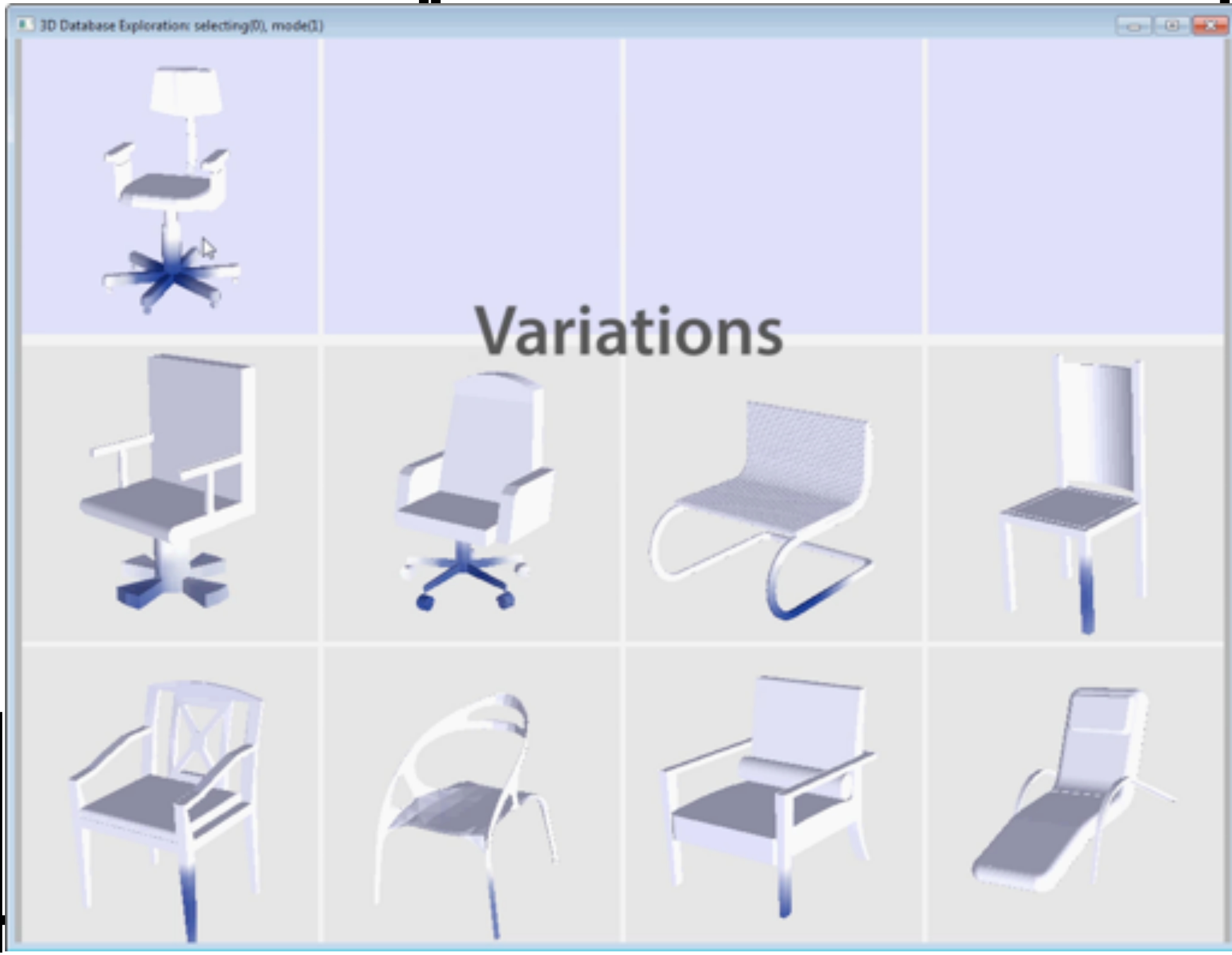
Grouping

Symmetry

Data Analysis

Exploration

Content Creation



Exploring Collections of 3D Models using Fuzzy Correspondences
V. Kim, W. Li, N. Mitra, S. DiVerdi, T. Funkhouser, SIGGRAPH'12

Recap + References

3D repositories

Structure

Correspondences

Parts

Variations

Grouping

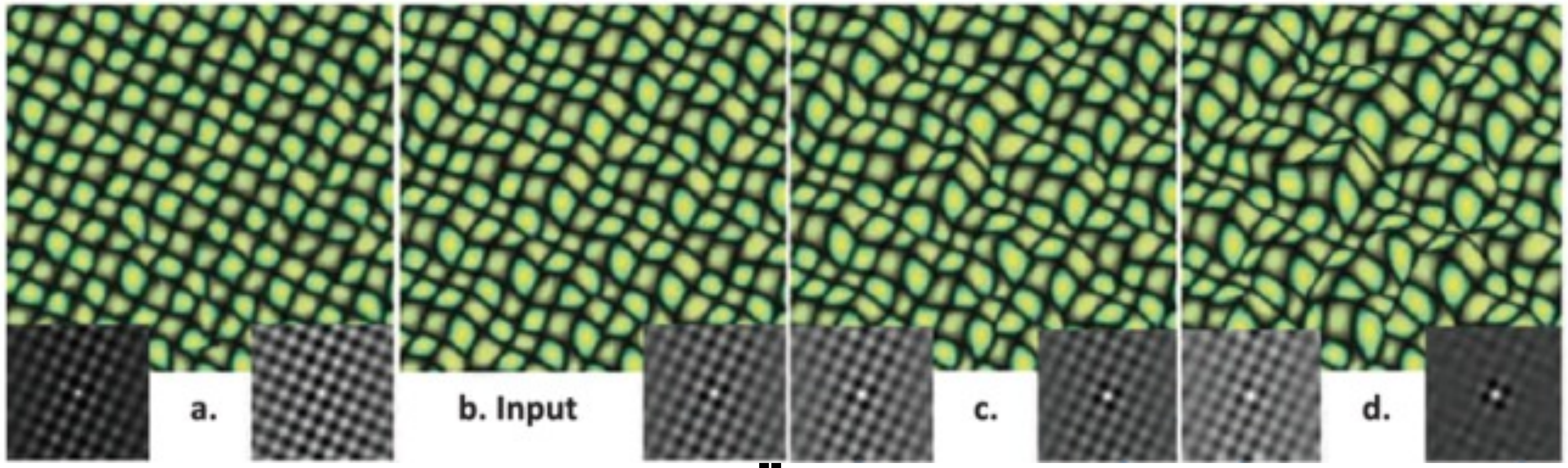
Symmetry

Data Analysis

Exploration

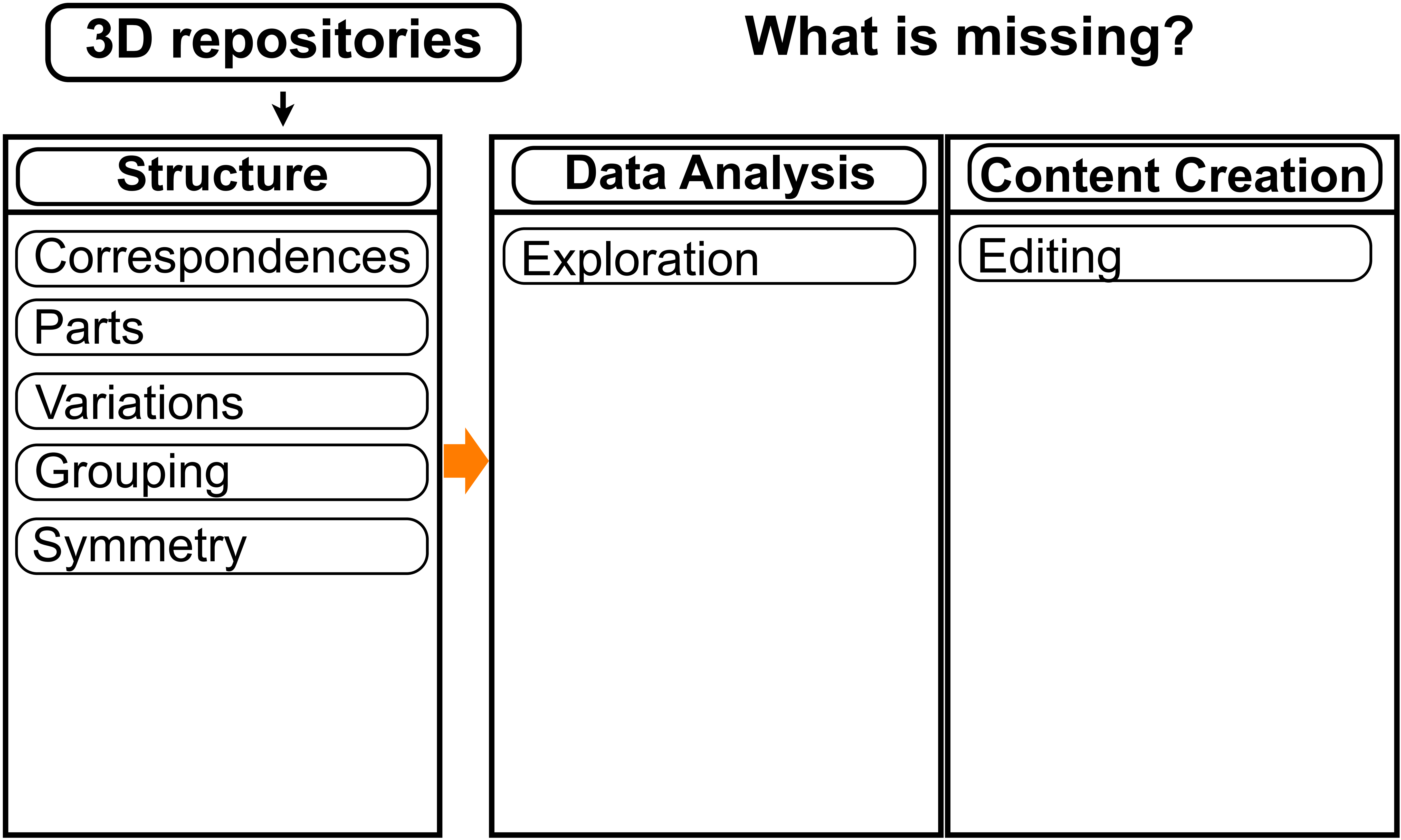
Content Creation

Editing



Symmetry-Guided Texture Synthesis and Manipulation
V. Kim, W. Li, N. Mitra, S. DiVerdi, T. Funkhouser,
Transactions on Graphics'12

Future Work



3D repositories

What is missing?

Structure

- Correspondences
- Parts
- Variations
- Grouping
- Symmetry

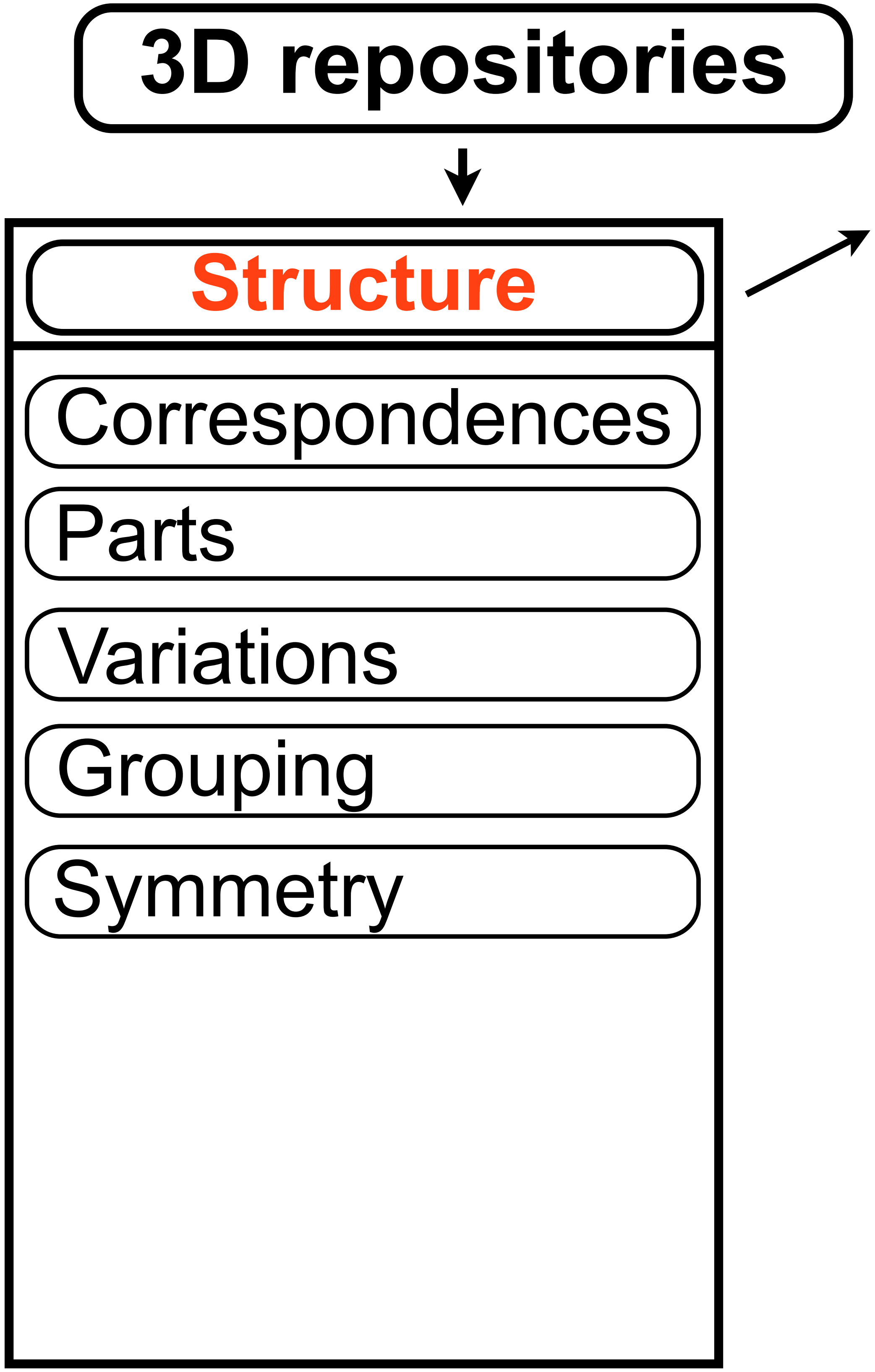
Data Analysis

- Exploration

Content Creation

- Editing

Future Work



3D repositories

Structure

Correspondences

Parts

Variations

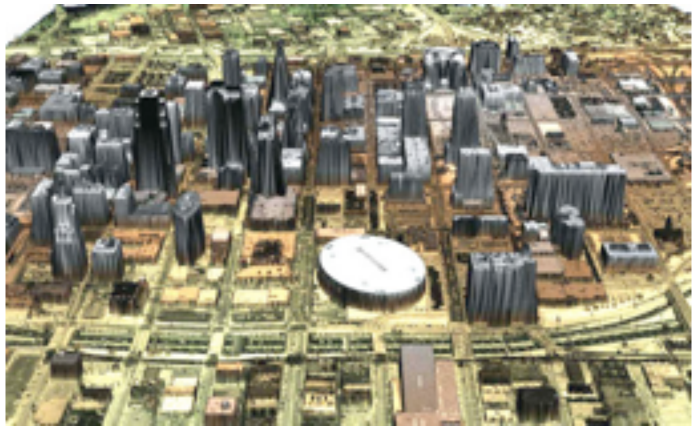
Grouping

Symmetry

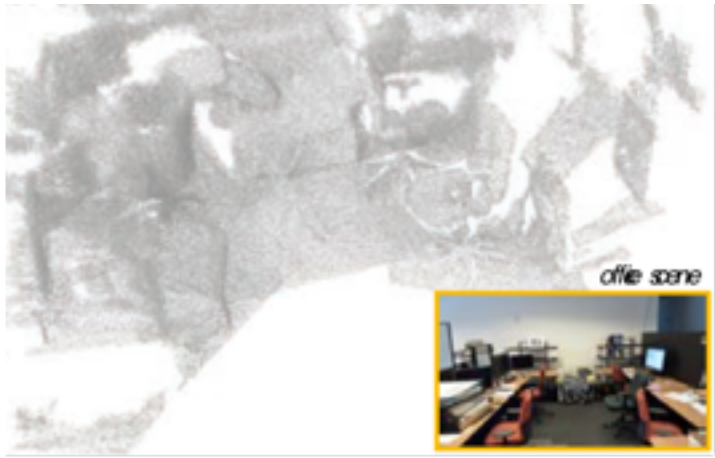
What is missing?

1. Define structure intrinsically:
- robust to articulation
- strong deformations

2. Define structure hierarchically:
- more efficient
- can handle more complex objects or scenes



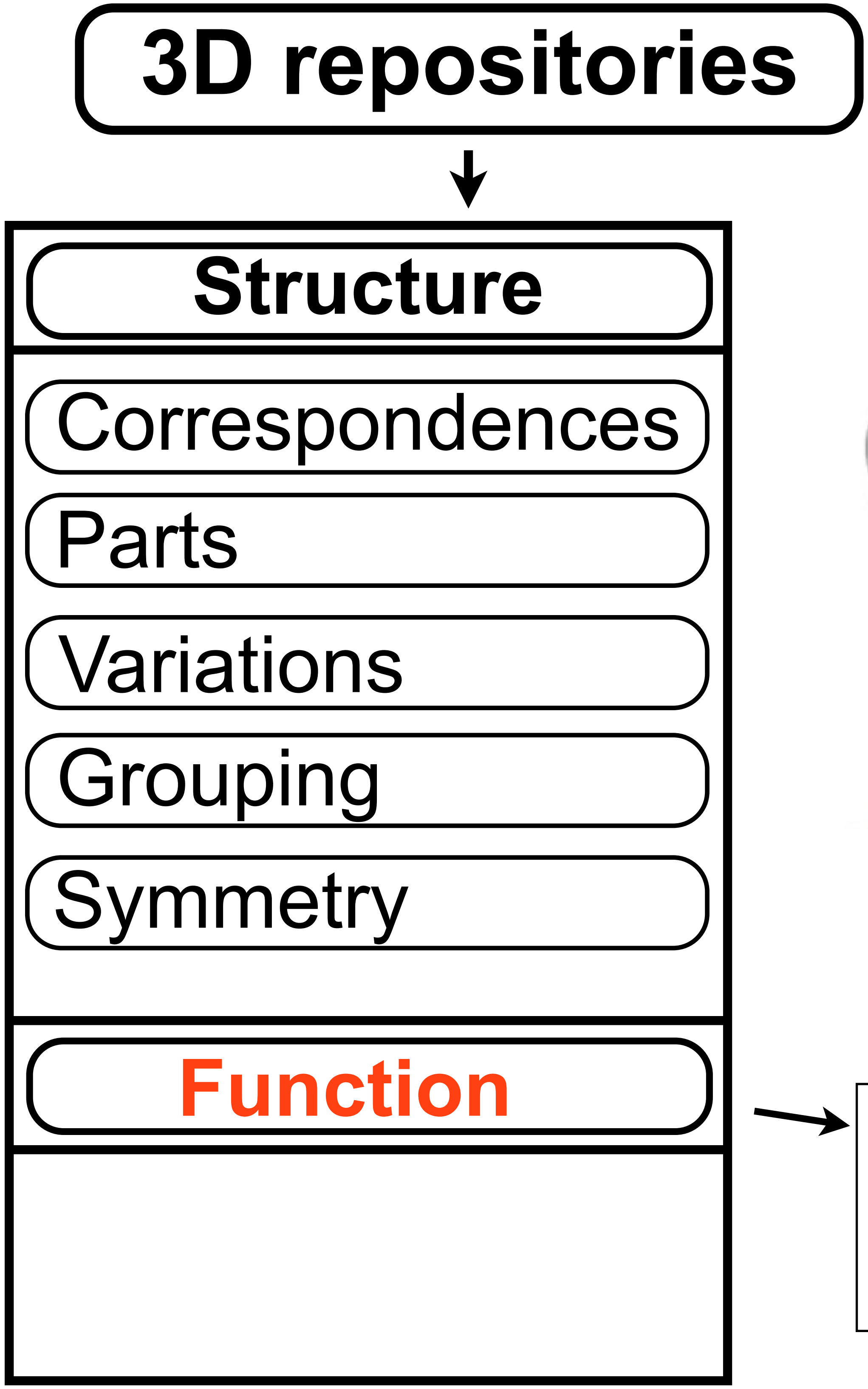
Google Streetview



input single-view scan

Microsoft Kinect

Future Work



3D repositories

Structure

Correspondences

Parts

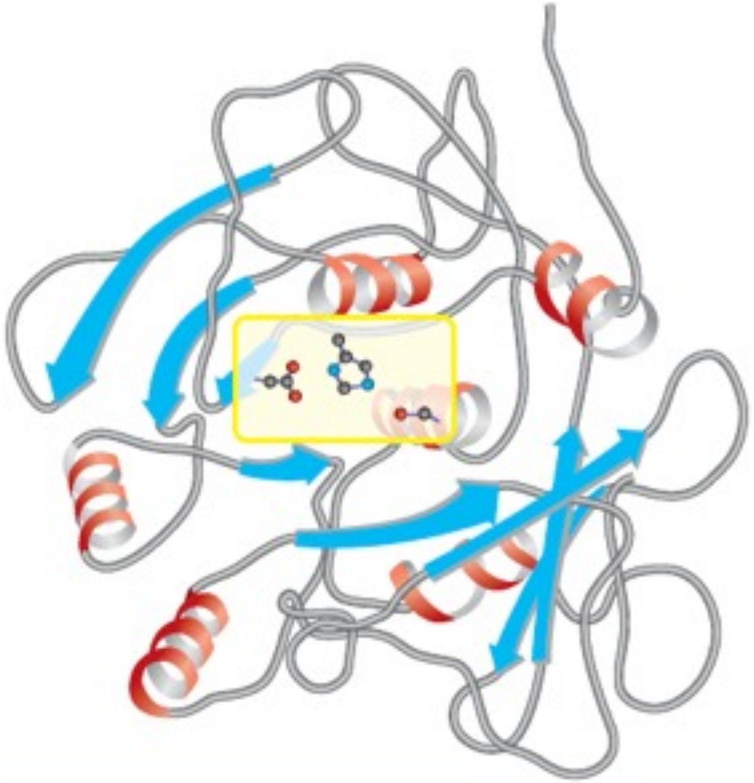
Variations

Grouping

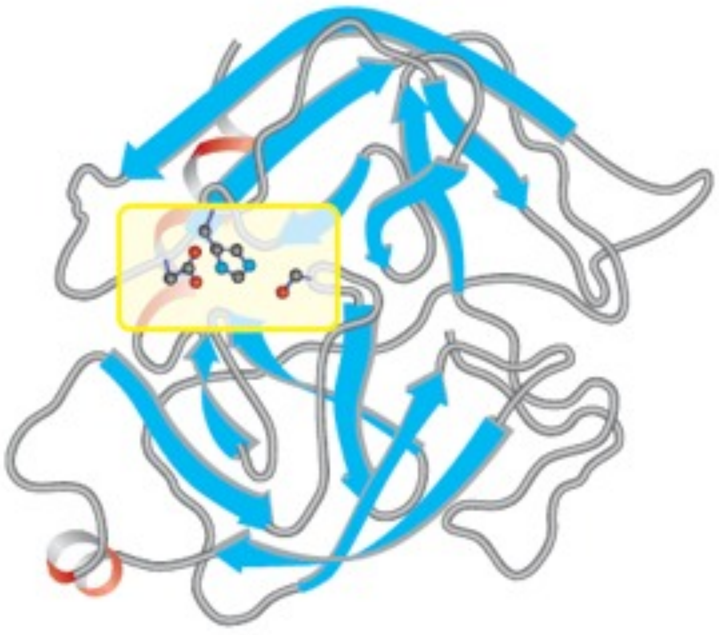
Symmetry

Function

What is missing?



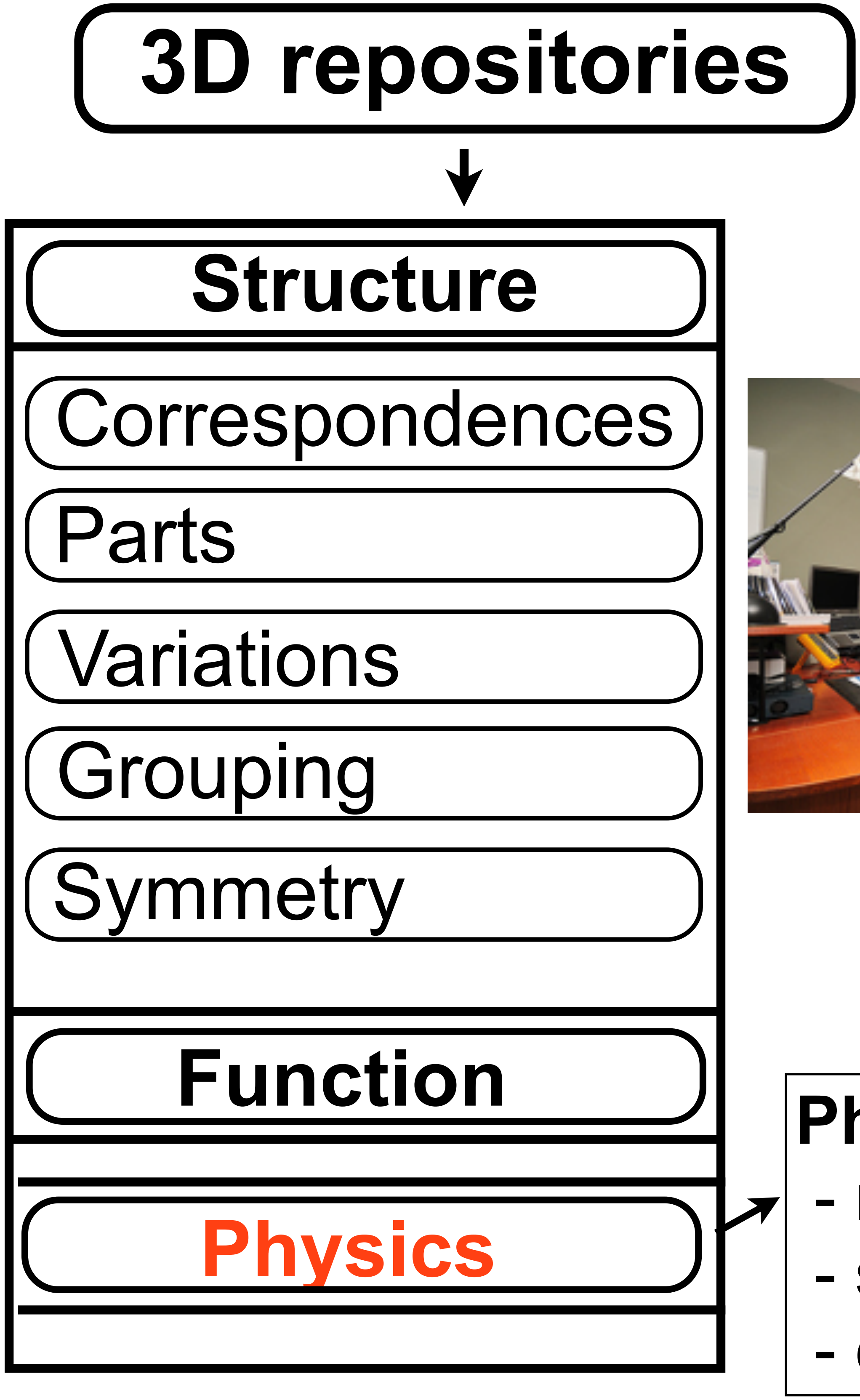
Subtilisin
(bacterial serine protease)



Chymotrypsin
(mammalian serine protease)

Relate function to geometry
- better analysis: know what is important
- interesting discoveries

Future Work



3D repositories

Structure

Correspondences

Parts

Variations

Grouping

Symmetry

Function

Physics

What is missing?

Flickr

Youtube

Physical properties of materials

- reflectance is important for 2D visual data
- stiffness is important for stability
- cost is important for manufacturing

Future Work

3D repositories

What is missing?

Structure

Correspondences

Parts

Variations

Grouping

Symmetry

Function

Physics

Data Analysis

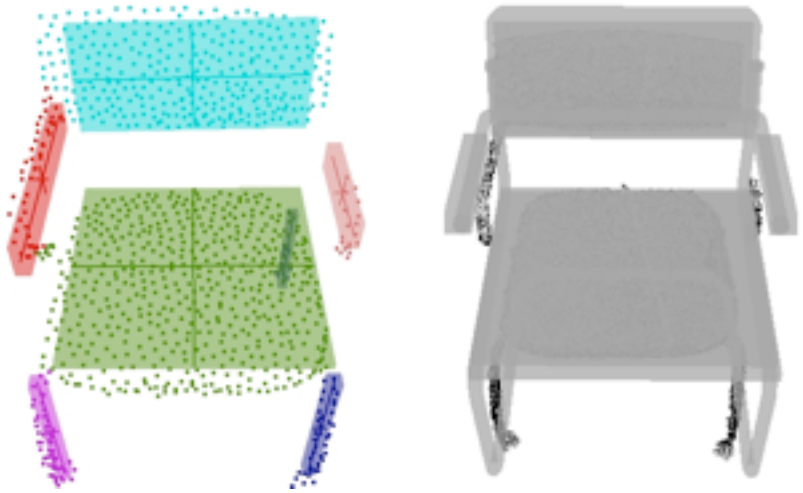
Exploration

Reconstruction

Content Creation

Editing

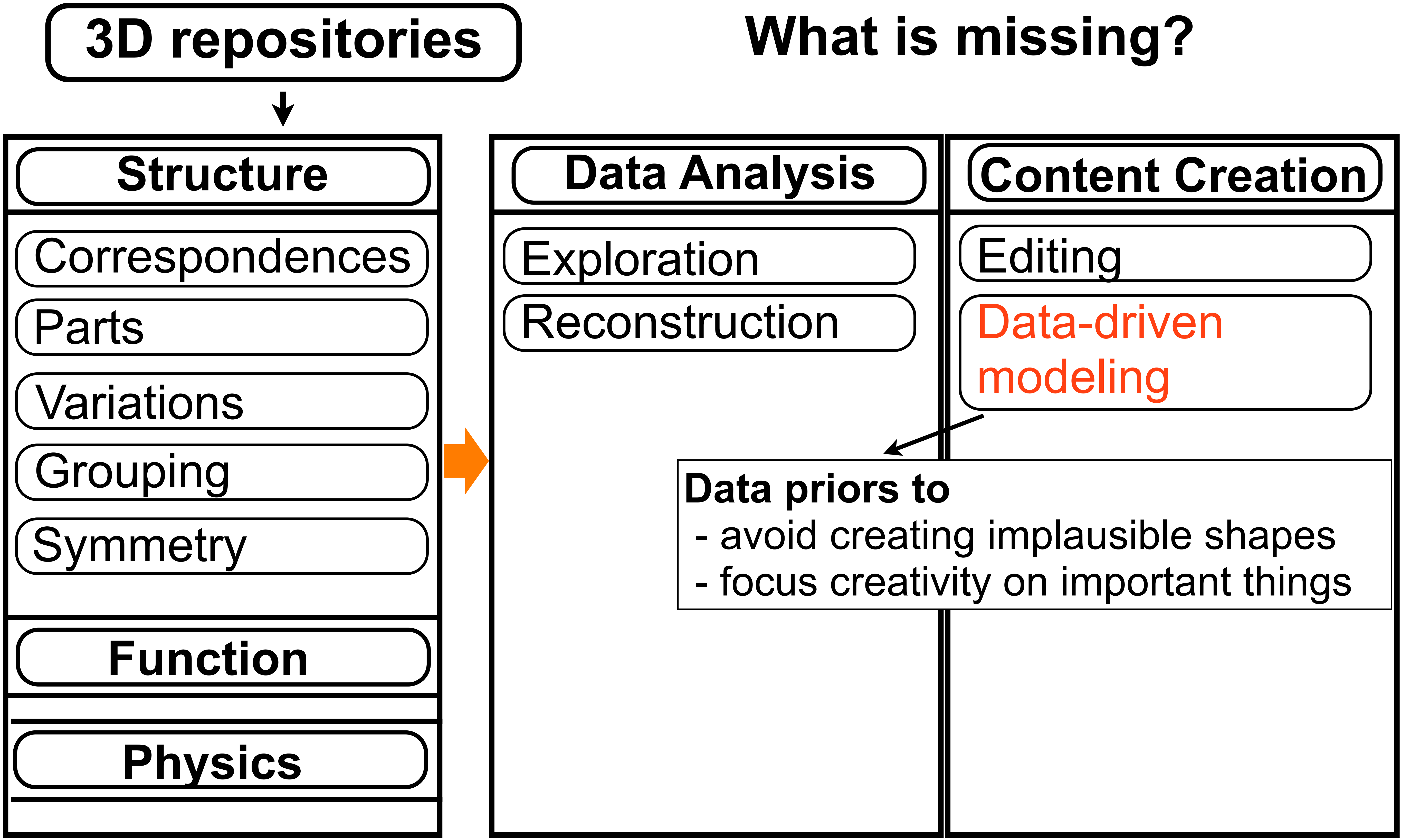
Data priors to
- resolve occlusions
- compensate for noise



a very preliminary example

Future Work

What is missing?



3D repositories

Structure

Correspondences

Parts

Variations

Grouping

Symmetry

Function

Physics

Data Analysis

Exploration

Reconstruction

Content Creation

Editing

Data-driven modeling

Data priors to

- avoid creating implausible shapes
- focus creativity on important things

Acknowledgement

Advisor: Thomas Funkhouser

Collaborators

- Siddhartha Chaudhuri, Xiaobai Chen, Stephen DiVerdi, Aleksey Golovinskiy, Wilmot Li, Yaron Lipman, Tianqiang Liu, Niloy J. Mitra

Princeton students and professors

Researchers who provided code and data

- Brown et al., Giorgi et al., Anguelov et al., Bronstein et al., Ovsjannikov et al., Nguyen et al., Huang et al., Wang et al.

Funding agencies:

- NSERC PGS-D, Siebel Scholarship, Adobe Internships
- NSF, AFOSR, Intel, Google, Adobe, Marie Curie CIG

Thank you!

ADDITIONAL SLIDES

Previous Work

3D repositories

↓

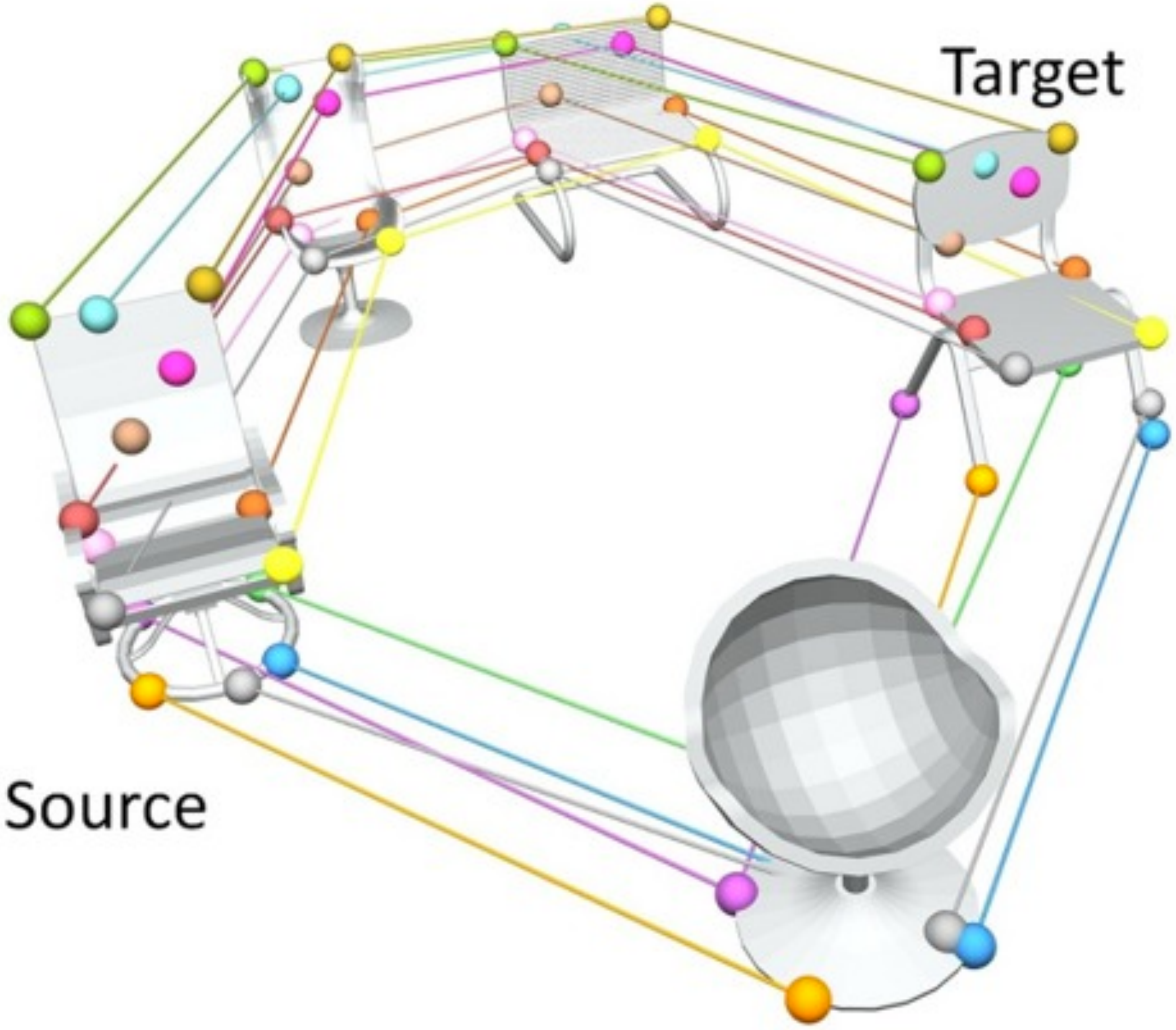
Structure

Correspondences

Parts

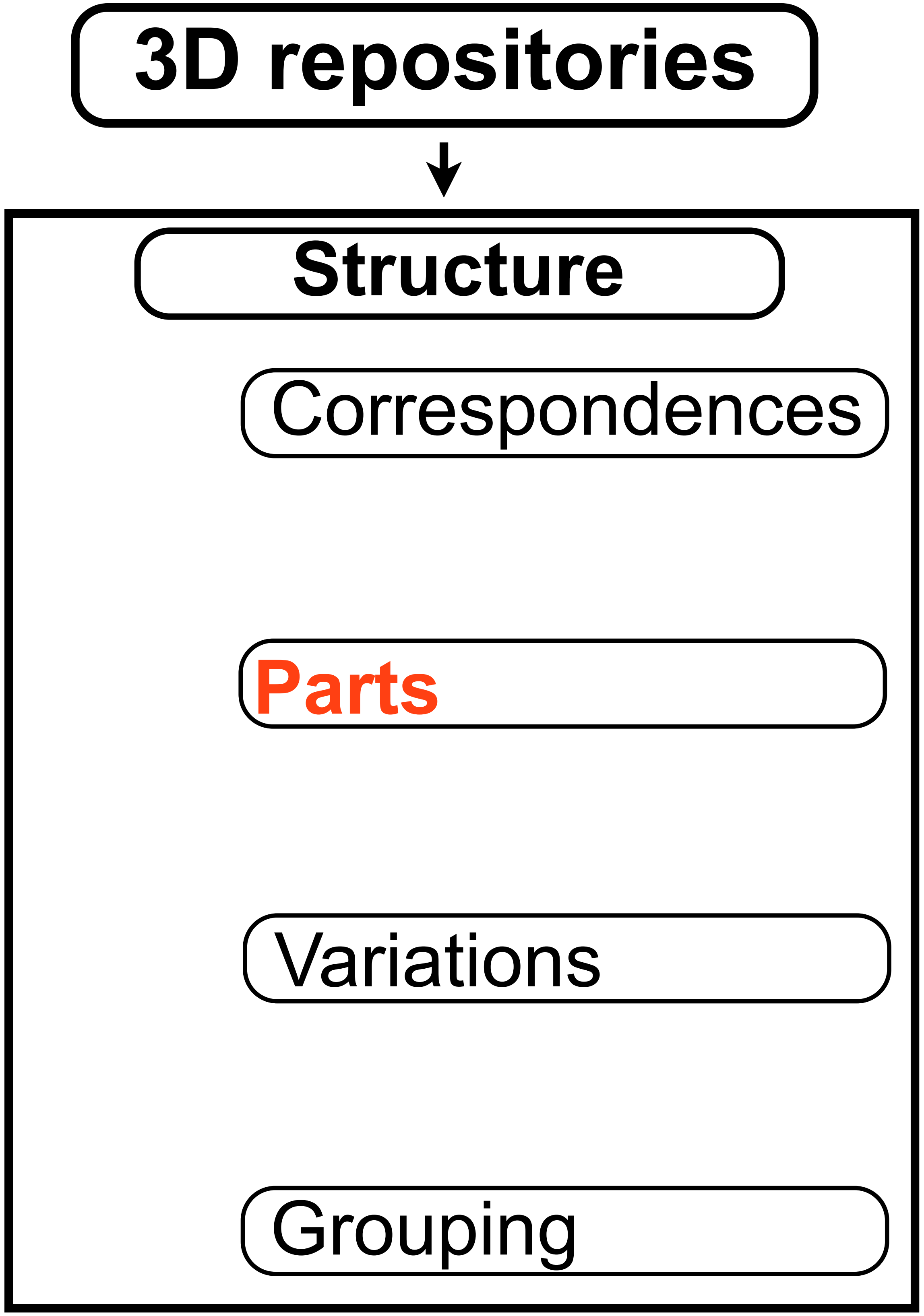
Variations

Grouping



(c) Huang et al. SIGGRAPH Asia'12

Previous Work



3D repositories

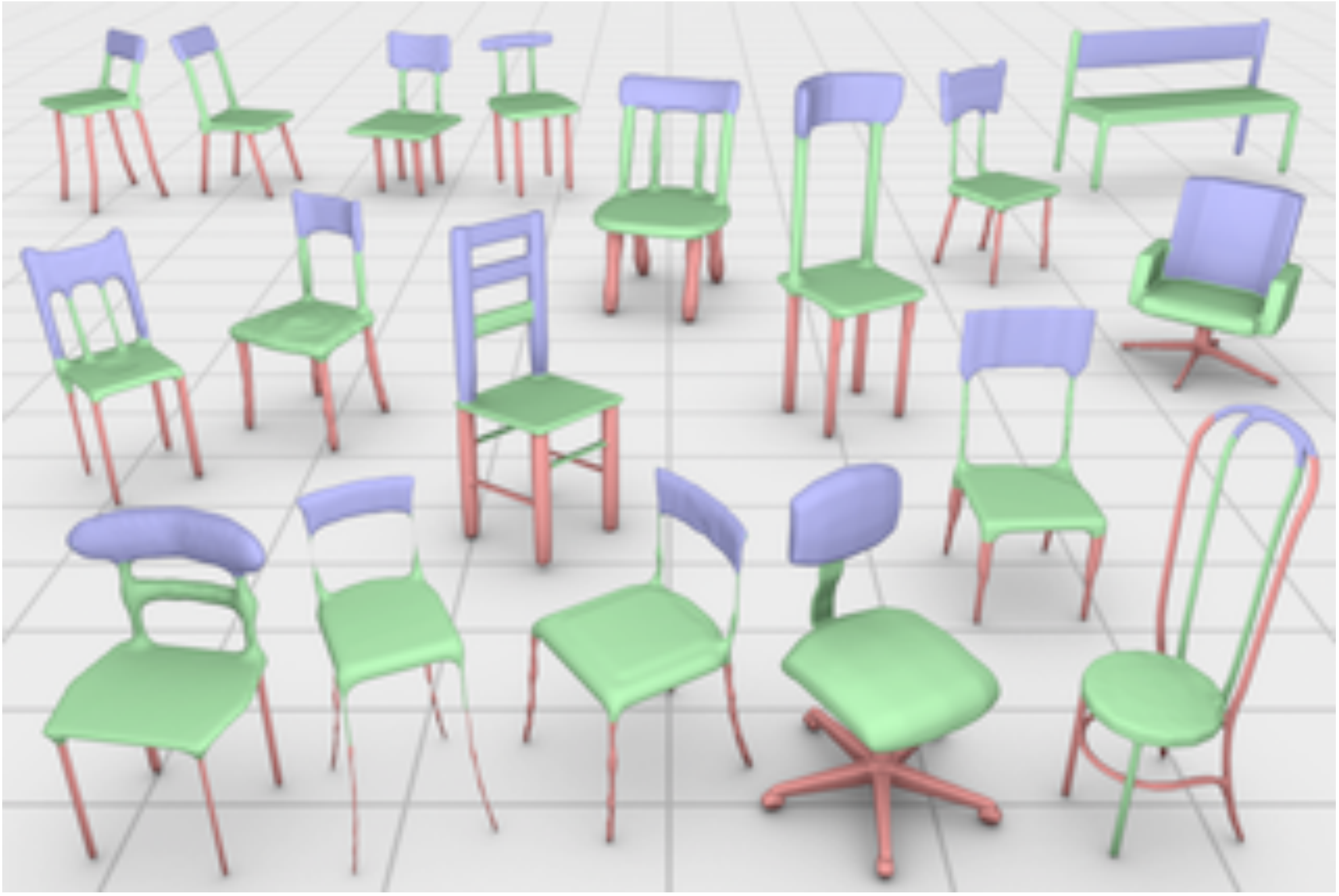
Structure

Correspondences

Parts

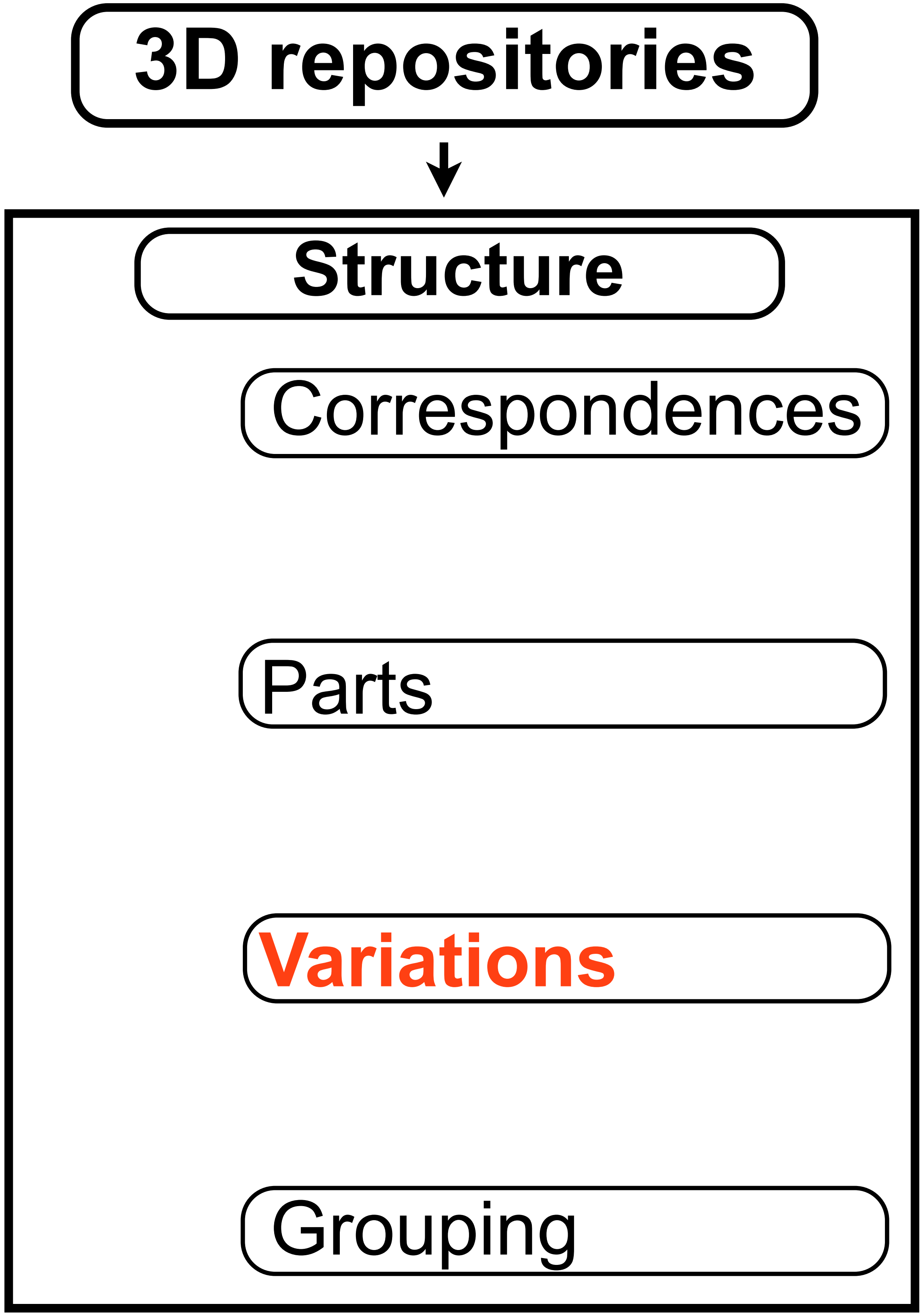
Variations

Grouping



(c) Sidi et al. SIGGRAPH Asia'11, Huang et al. SIGGRAPH Asia'11

Previous Work



3D repositories

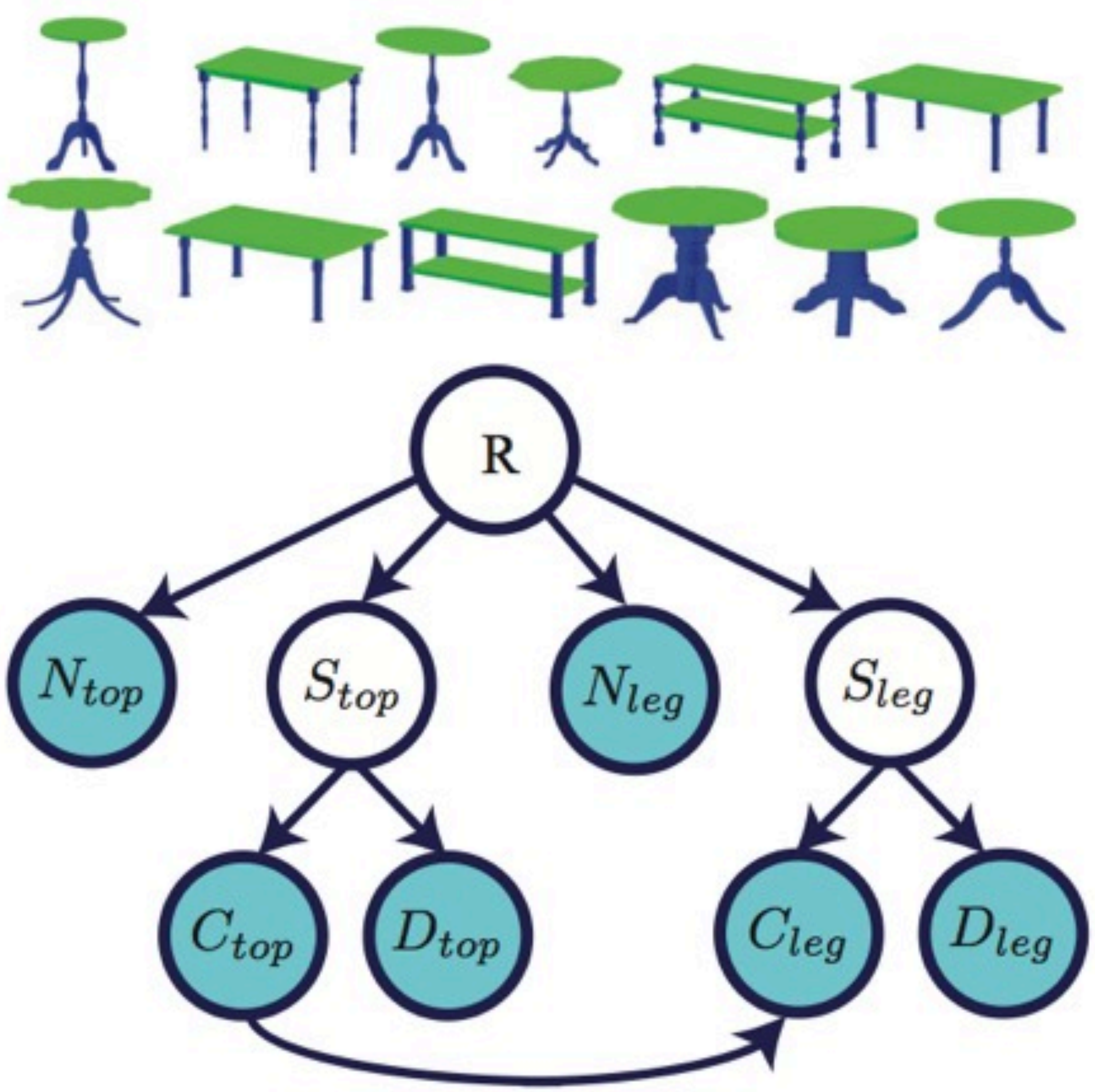
Structure

Correspondences

Parts

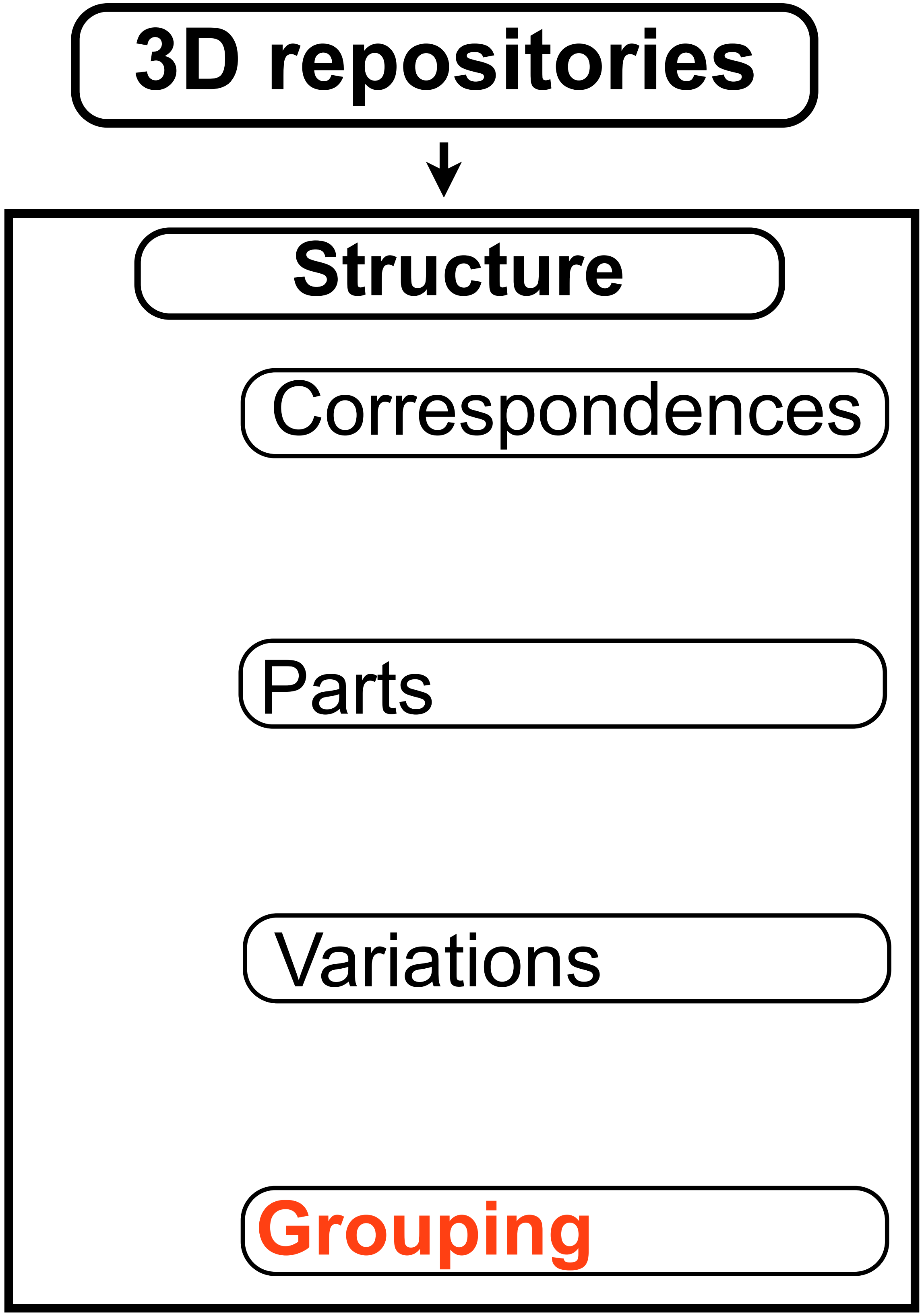
Variations

Grouping



(c) Kalogerakis et al., SIGGRAPH’12

Previous Work



3D repositories

Structure

Correspondences

Parts

Variations

Grouping

Gears

Screws

(c) Hou et al., CAD’05

Recap + References

3D repositories

Structure

Correspondences

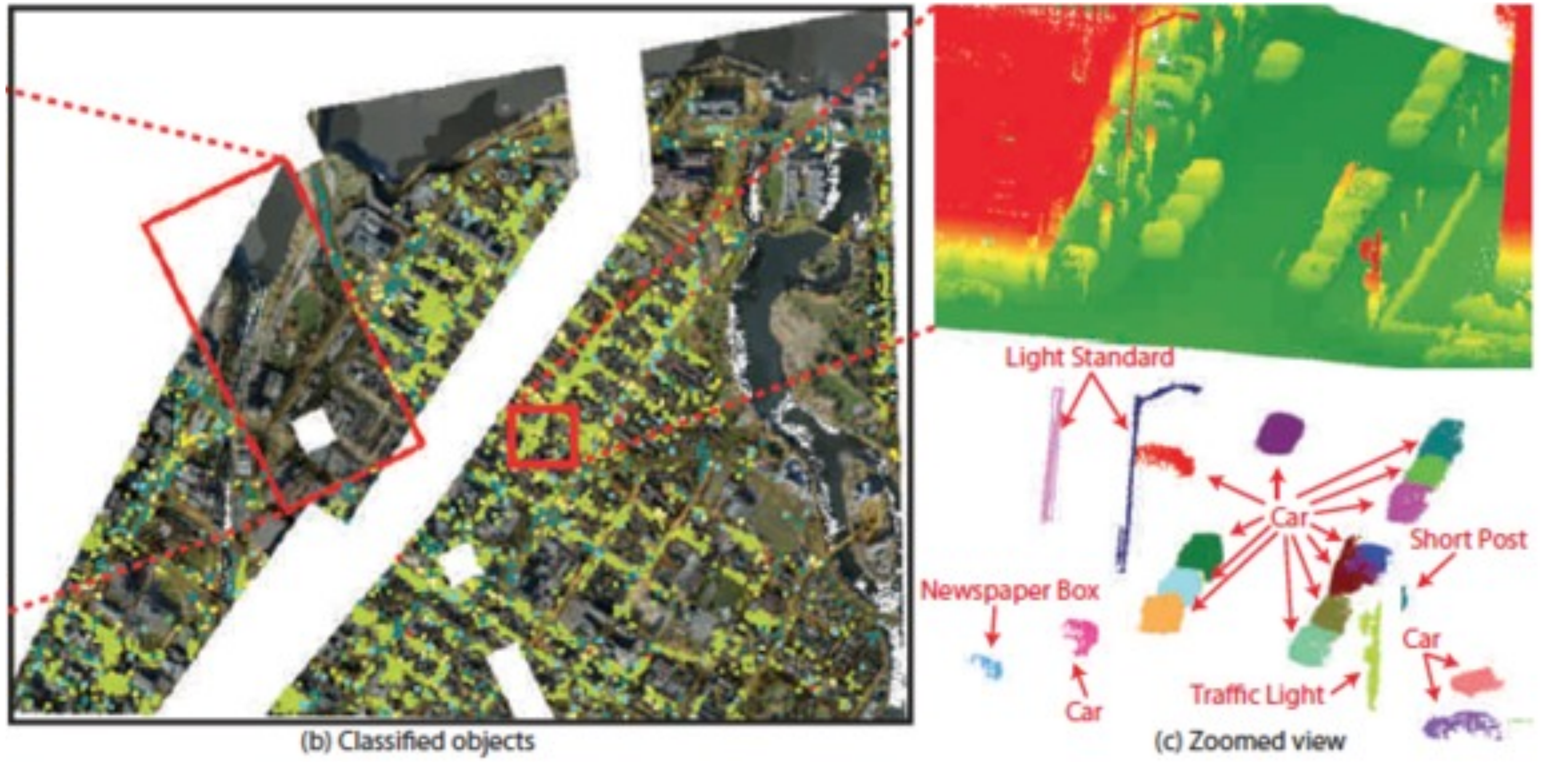
- Non-isometric shapes
- Leverage power of the set

Parts

Variations

Grouping

Symmetry



(b) Classified objects

(c) Zoomed view

Shape-based Recognition of 3D Point Clouds in Urban Environments
A. Golovinskiy, V. Kim, T. Funkhouser, ICCV’09